CDR Update Workshop

Eddy Grąbczewski

e.grabczewski@rl.ac.uk

Database Design Engineer

Rutherford Appleton Laboratory

18 Jan 2006

Introduction

In these three workshops we shall:

1. *Review* CDR Data Model Features.
2. *Highlight* CDR Data Model Changes.
3. *Recommend* CDR Data Modelling Best Practices.

1. CDR Data Model Features

1. CDM architecture:
2. Object Class Hierarchy
3. Entity and Object data modelling
4. Object Classes and Entity Roles
5. Relationship Entities:
6. Network Hierarchy support:
7. Valid Time support:
8. Transaction Time support:
9. Distributed database rows:
10. Multiple Language support:
11. Multiple Entity Signs support:
12. ISO Standard SQL3 attribute data types
13. Glossary of table and attribute names & abbreviations
14. Colour coded data model layout
15. Object row visibility and Object row order support

CDM Architecture

The Corporate Data Repository (CDR) hosts the Corporate Data Model (CDM).

The CDM architecture is based on the following:

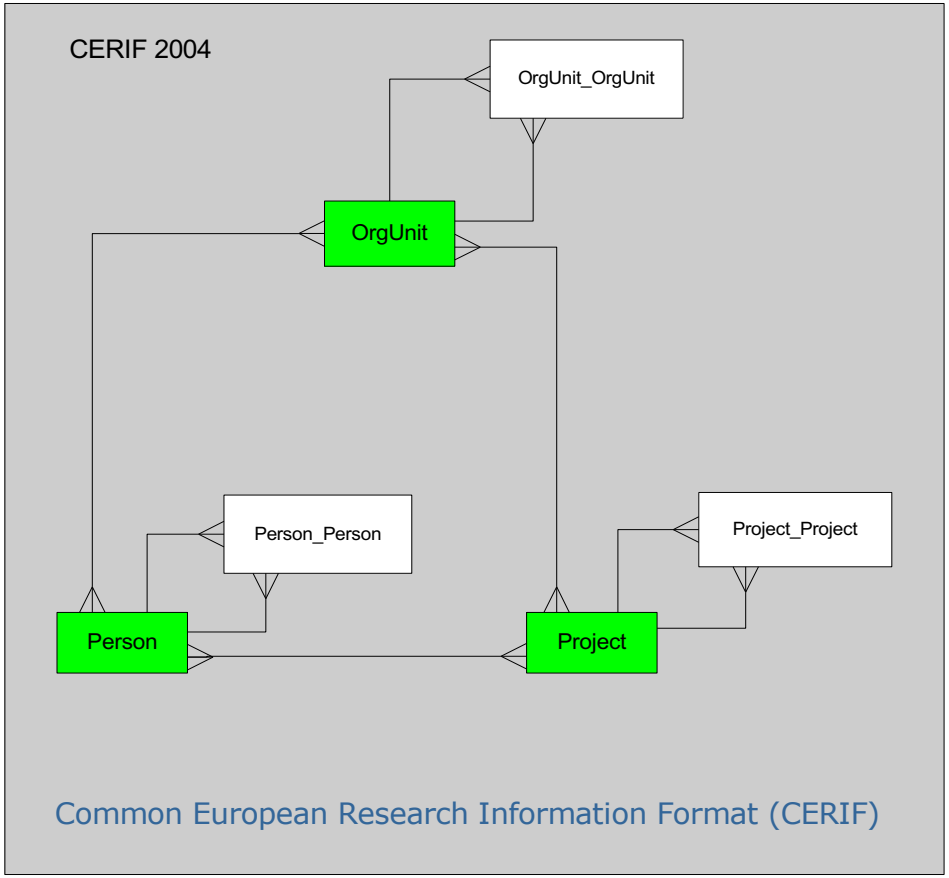
- CERIF Architecture.
- ANSI/X3/SPARC Architecture

CERIF Architecture

The CERIF Architecture comprises:

- Three Base entities
- Fully-connected relationships between the Base entities
- Recursive, many-to-many relationships on each Base entity

Conformance to the CERIF 2004 standard was a design constraint placed on the CDR design by Keith Jeffery by May 2003.

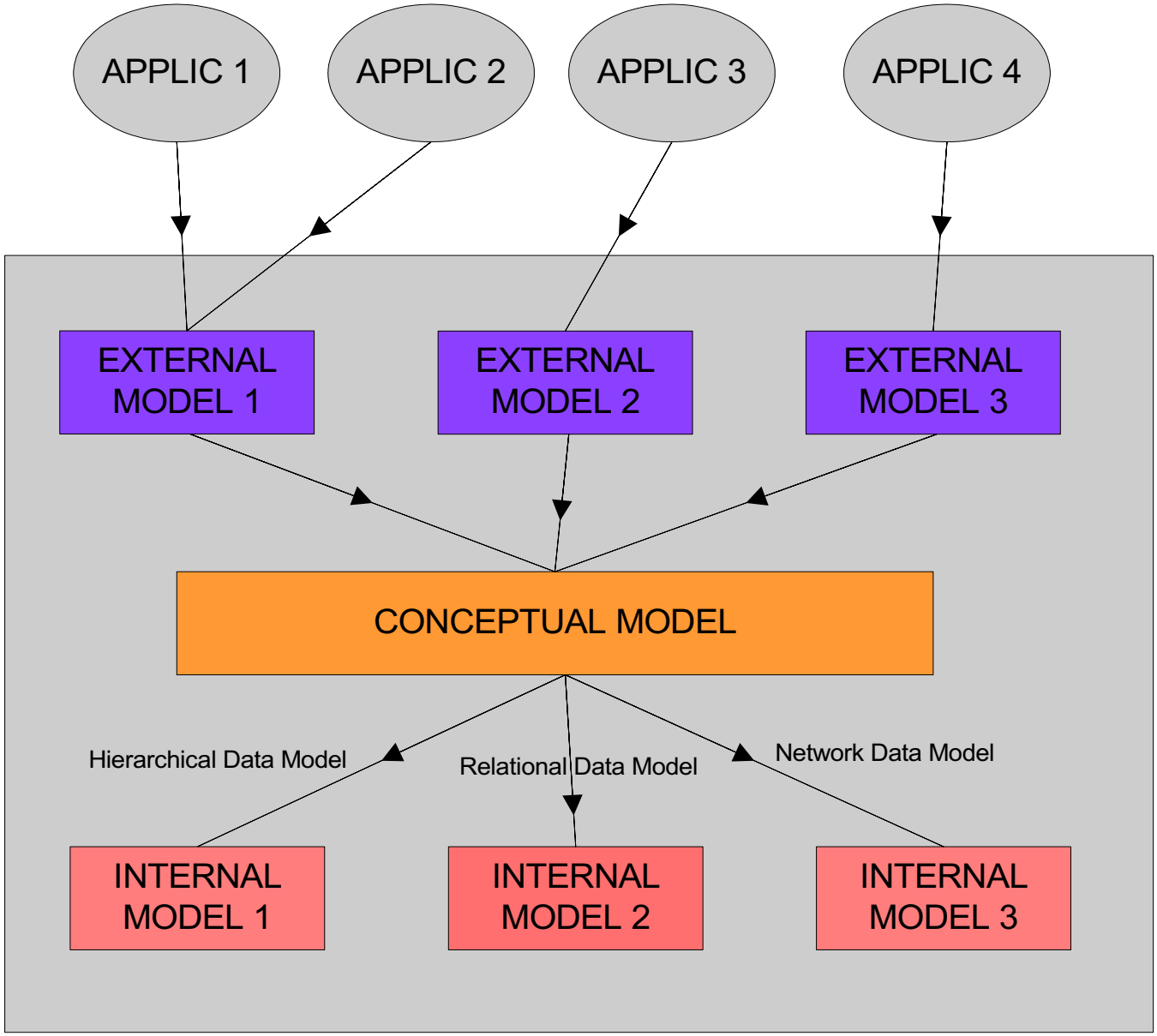


ANSI/X3/SPARC Architecture

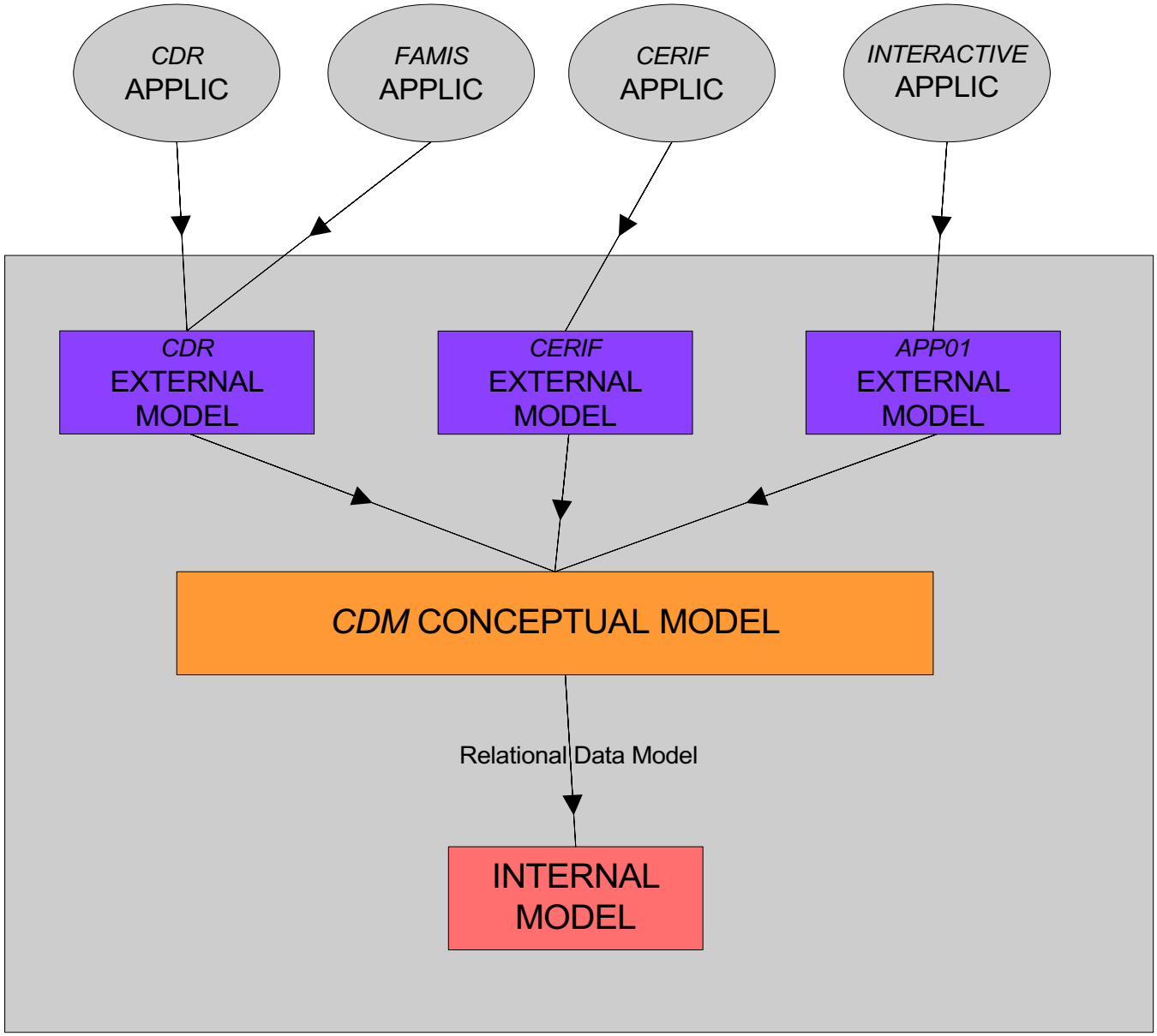
The *Interim Report* (1975) of the *ANSI/X3/SPARC* Study Group on DBMSs also proposed three "realms of interest" in a database architecture:

- External realm: a simplified model of the real world as seen by *one or more* applications.
- Conceptual realm: a limited model of the real world for *all* applications.
- Internal realm: data in computer storage representing the *conceptual realm*.

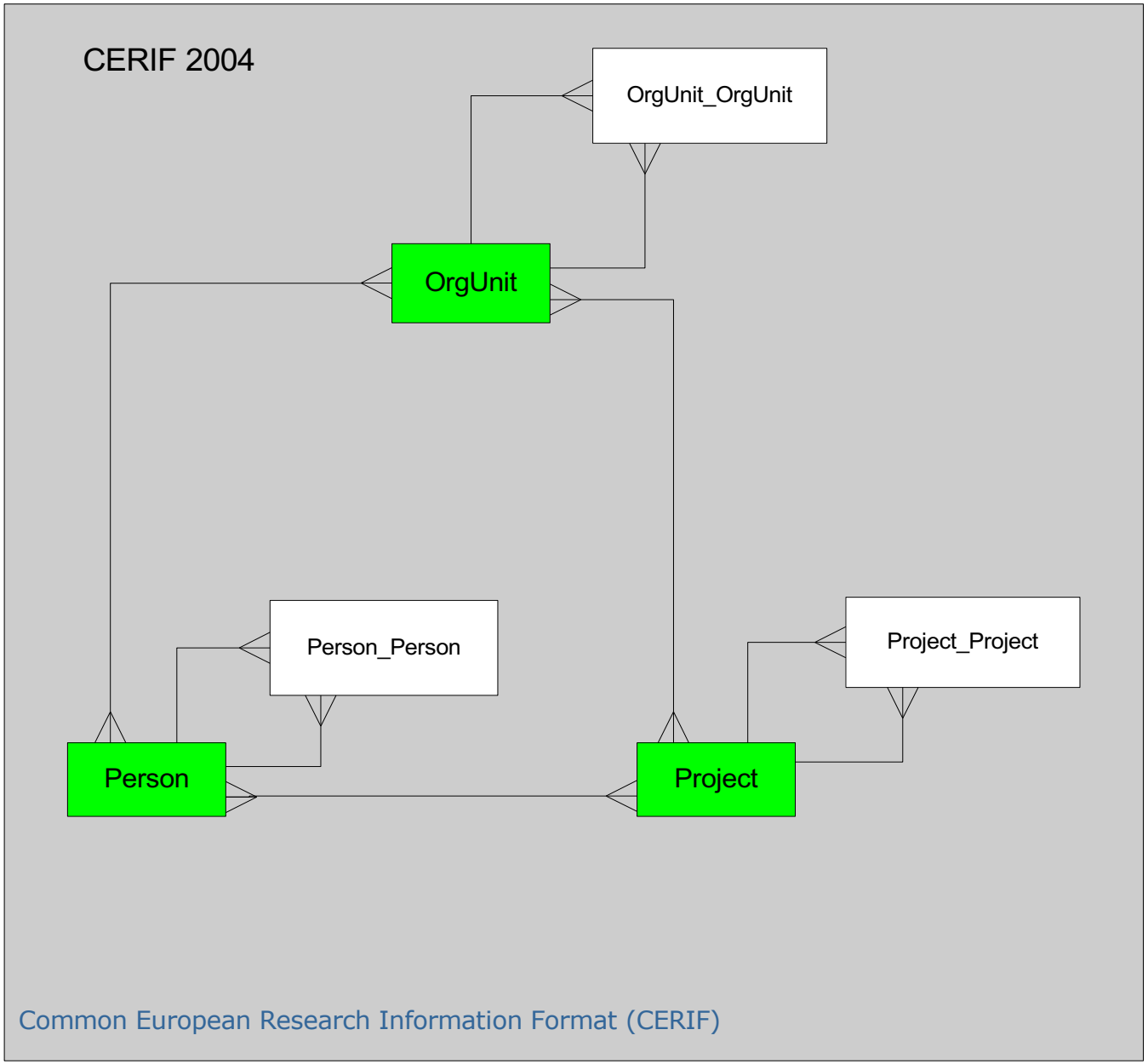
ANSI/X3/SPARC Architecture



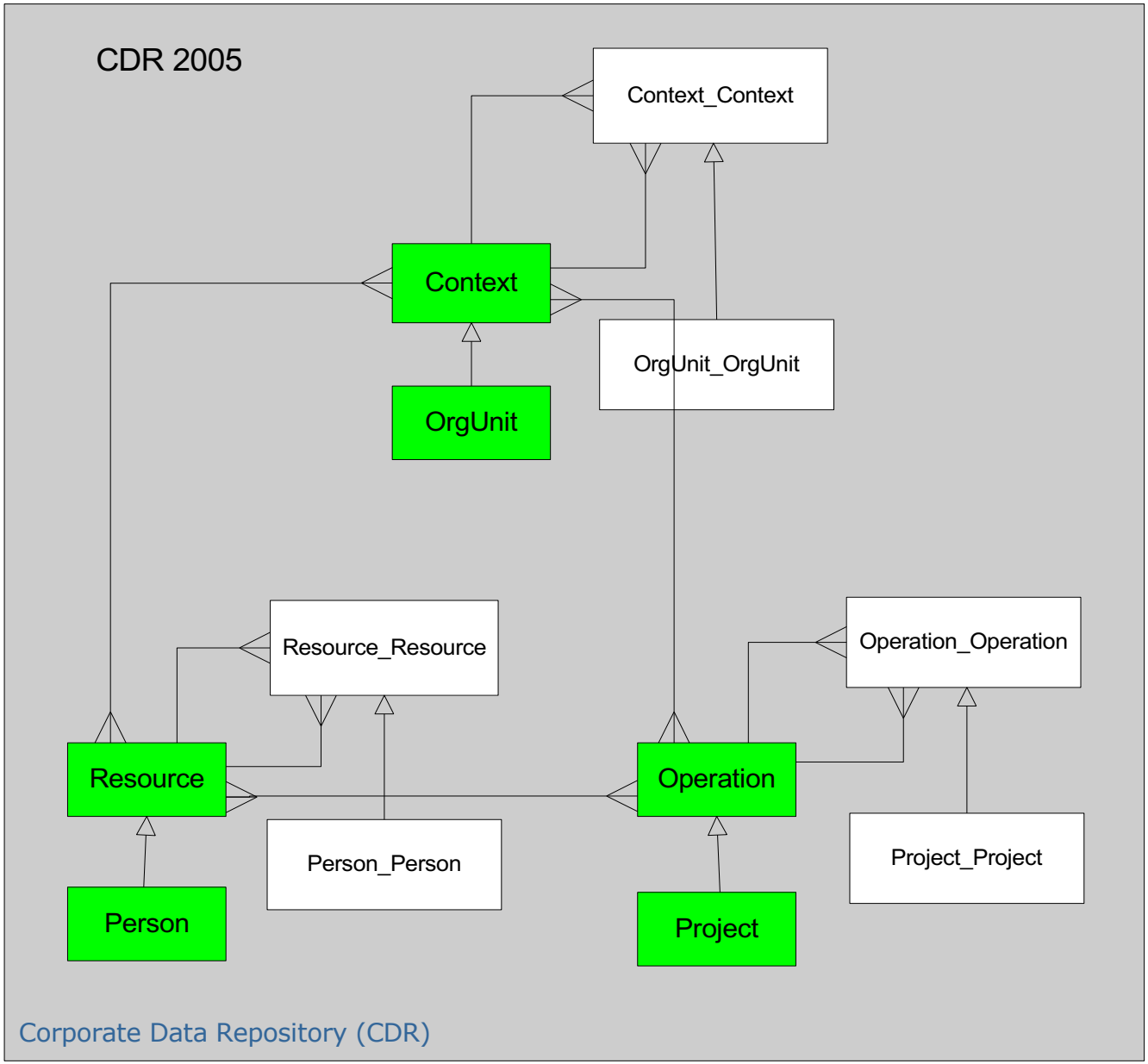
CDM Architecture



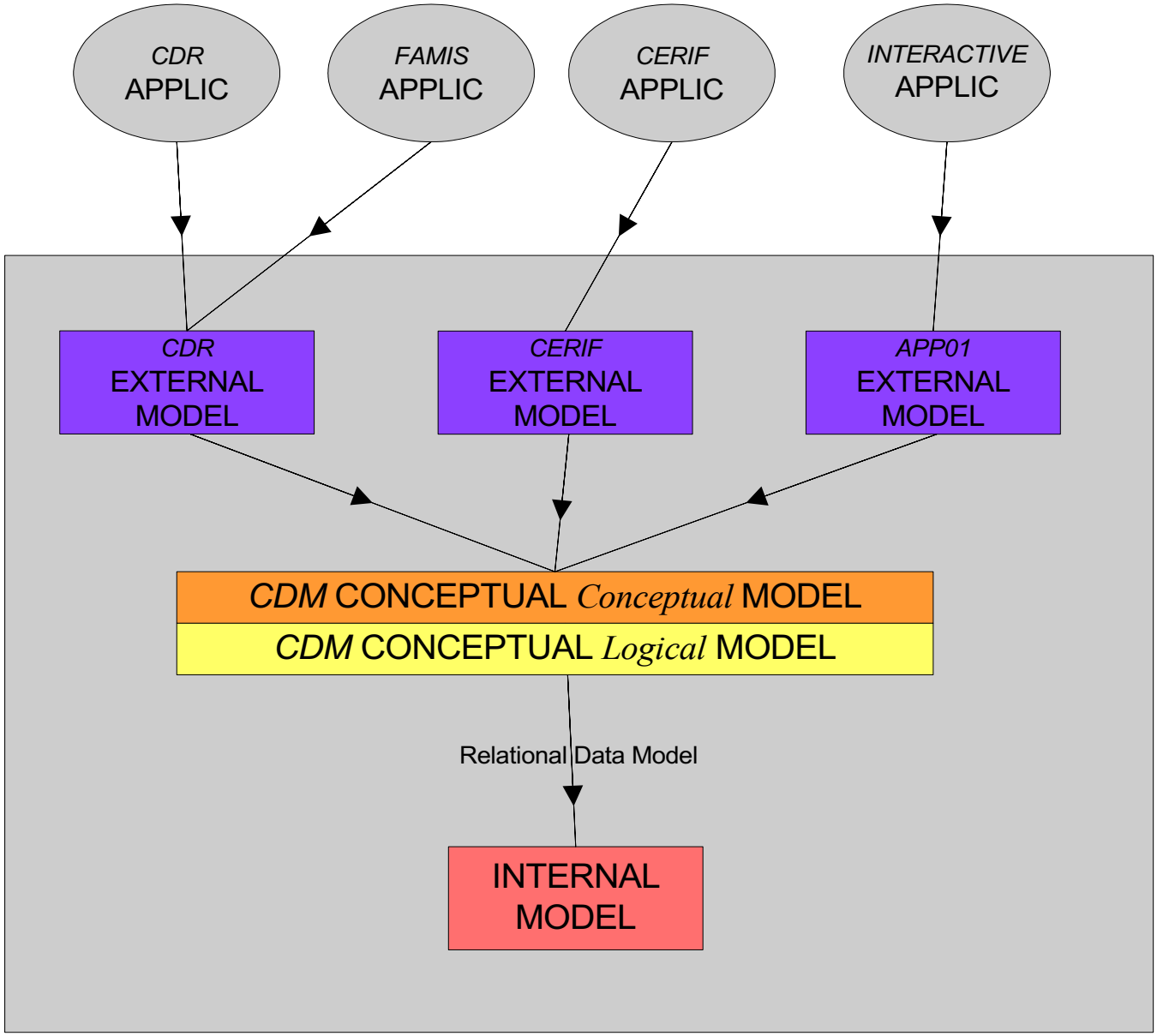
CDM External Model : CERIF 2004



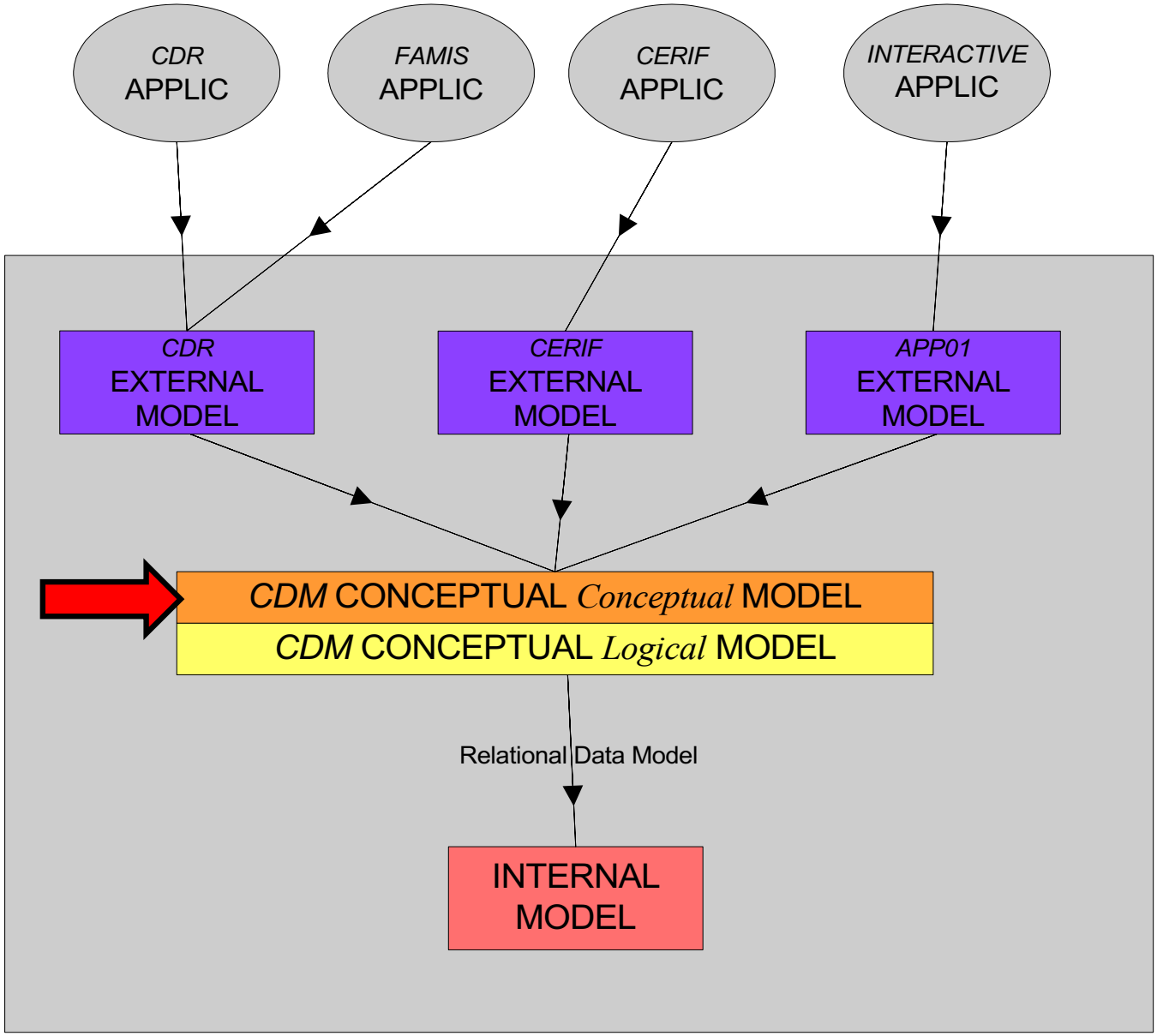
CDM External Model: CDR 2005



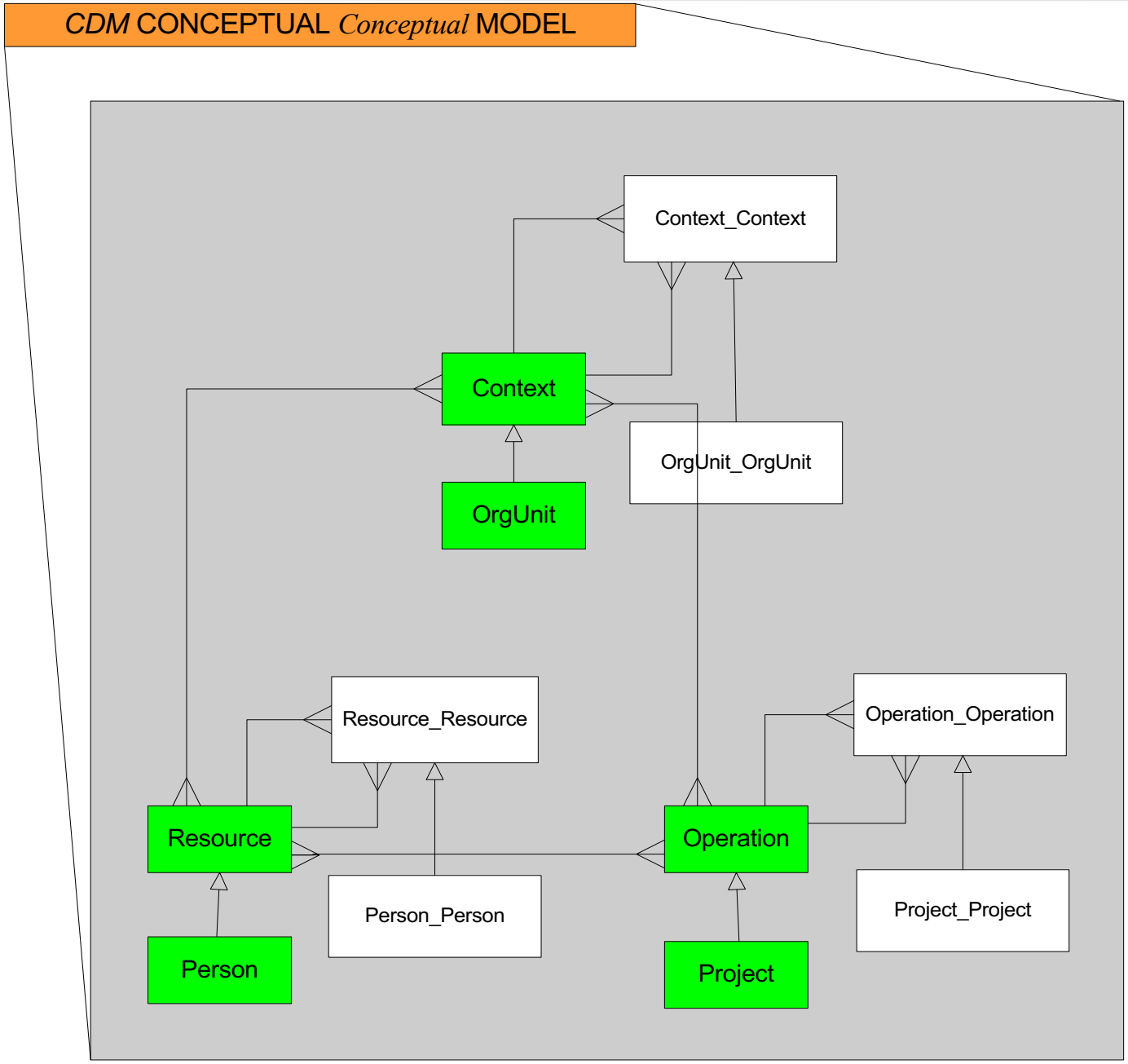
CDM Conceptual Model *Sub-levels*



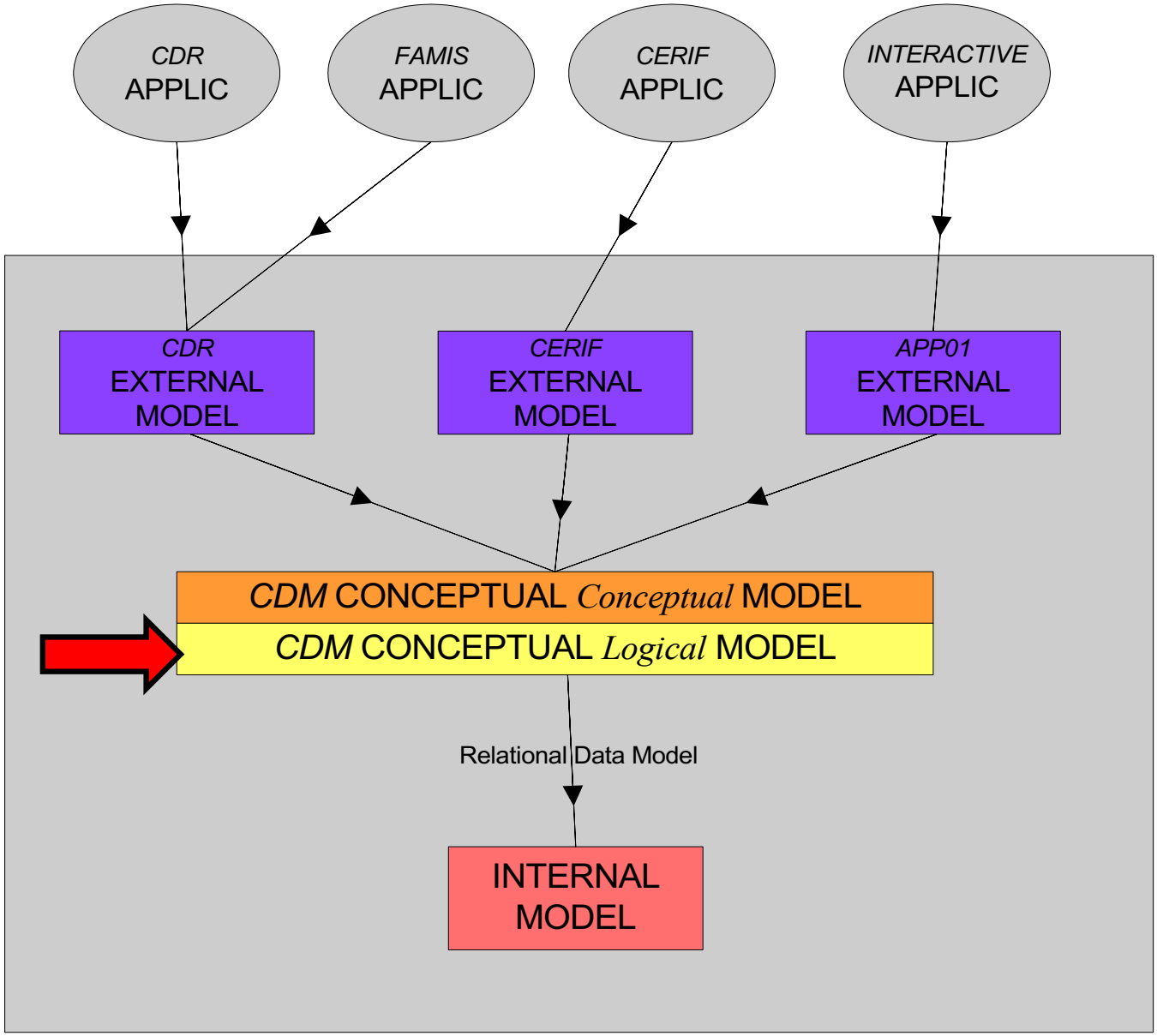
CDM Conceptual *Conceptual* Model



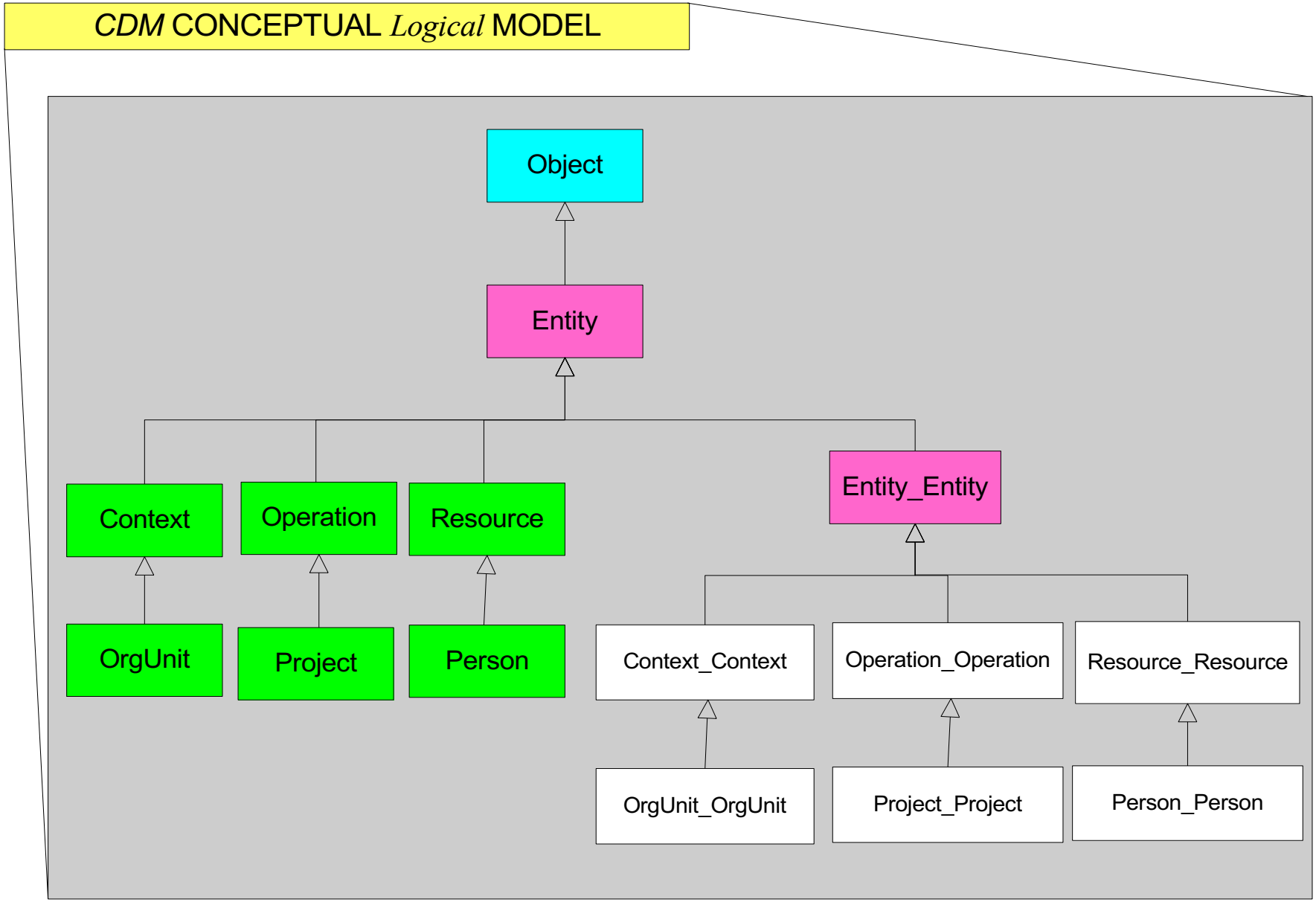
CDM Conceptual *Conceptual* Model



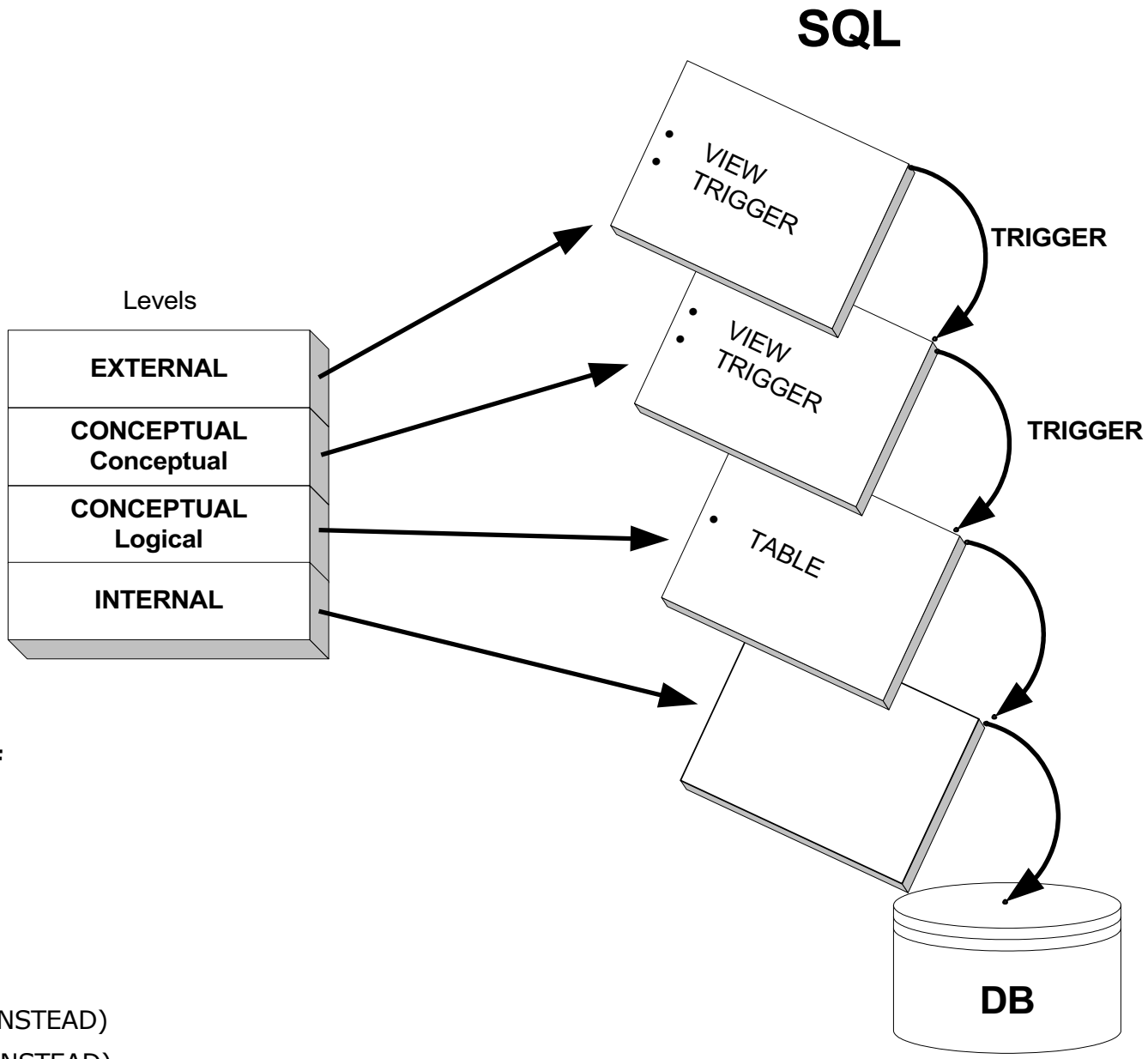
CDM Conceptual *Logical* Model



CDM Conceptual *Logical* Model



CDM Architecture - Implementation



CREATE TRIGGER … INSTEAD OF

Supported By:

- Oracle
- Microsoft SQL Server
- IBM DB2
- IBM Informix
- CA Ingres (CREATE RULE … DO INSTEAD)
- PostgreSQL (CREATE RULE … DO INSTEAD)

CCLRC

Object Class Hierarchy

- The *Object Class Hierarchy* comprises a set of entities (tables) at the Conceptual *Logical* (CL) level of the data model.
- This is sometimes call this an *ontology*.
- This class structure represents the way in which we *fundamentally* agree to conceptualise the business worlds of both *CCLRC* and *Diamond*.

CDM Entities and Objects

- The *Interim Report* (1975) of the *ANSI/X3/SPARC* Study Group on DBMSs stated the following:

  "*An object in the real world is called an entity*"

- A paper by *Hall, Owlett* and *Todd* (1976) discusses the concept of a *surrogate*:

  "… *every entity of the outside world is associated with a surrogate which stands for that object in the model*"

- A *surrogate* is an object in the model that is in one-to-one correspondence with an entity in the world

CDM Entity and Object Identifiers

Entity Identifier

- an identifier used to identify the unique existence of an *entity* in the real world.
- a *surrogate*.

Object Identifier

- an identifier used to identify the unique existence of an *object* in the database.
- a global *primary key*.

NOTE
An Entity Identifier may have several Object Identifiers.

erPerson

ObjId	ObjDomId	EntId	EntDomId	ValTimeBegin	ValTimeEnd	PerFamilyNames	PerFirstNames
1000100	9000000	1000081	9000000	01 Jan 995 00:00	12 Nov 1035 00:00	Canute	King
1000101	9000000	1000083	9000000	01 Apr 1995 09:00	18 Sep 2002 09:00	Smith	Mary
1000105	9000000	1000083	9000000	19 Sep 2002 09:00	NULL	Jones	Mary
1000107	9000000	1000089	9000000	01 Apr 1995 09:00	NULL	Baker	Benjamin

CDM Entity and Object Domains

Each Identifier belongs to a corresponding Domain:

Entity Domain
: the *entity* domain to which an Entity Identifier belongs.

Object Domain
: the *object* domain to which an Object Identifier belongs.

ccPerson

ObjId	ObjDomId	EntId	EntDomId	PerFamilyNames	PerFirstNames
1000100	9000000	1000081	9000000	Canute	King
1000101	9000000	1000083	9000000	Smith	Mary
1000105	9000000	1000083	9000000	Jones	Mary
1000107	9000000	1000089	9000000	Baker	Benjamin John
1000999	6000000	1000089	9000000	Baker	Benjamin

CDM Entity and Object Domains

Why do we need two domains?

In the earlier data model we only had one domain called DatabaseId.

CCLRC

Relationship Entities

1. Simple Relationship Entity
   - a relationship between only *Independent Entities*
2. Complex Relationship Entity
   - a relationship between any mix of *Independent Entities*, *Simple Relationship Entities* or *Complex Relationship Entities*.

Network Hierarchy support

The following functions and procedures implement a solution to *The Parts Explosion* problem: how to find the components of any given part, given any table P and P_P.

EXPLODE procedures and functions:

Functions:

cfEXPLODE

cfREFLEXIVE_EXPLODE

Procedures:

cpEXPLODE

cpREFLEXIVE_EXPLODE

TRANSITIVE CLOSURE procedures and functions:

Functions:

cfTRANSITIVE_CLOSURE

cfREFLEXIVE_TRANSITIVE_CLOSURE

Procedures:

cpTRANSITIVE_CLOSURE

cpREFLEXIVE_TRANSITIVE_CLOSURE

Network Hierarchy support: EXPLODE

FUNCTION cfEXPLODE - the function returns a table of values:

```
SELECT *
FROM TABLE(cfEXPLODE('erContext_Context',
                     'CntxEntId1', 'CntxEntDomId1',
                     'CntxEntId2', 'CntxEntDomId2')) EXCC;
```

PROCEDURE cpEXPLODE – to see the results of the procedure call, query the ciEXPLODE table:

```
BEGIN
 cpEXPLODE('erContext_Context',
           'CntxEntId1', 'CntxEntDomId1',
           'CntxEntId2', 'CntxEntDomId2');
END;
/
SELECT * FROM ciEXPLODE;
```

Network Hierarchy support: EXPLODE

INPUT

ENTID	ENTDOMID	ENTROLENAME	CNTXNAME	CNTXSHORTNAME
1000004	6000000	Organisation	CLRC	CLRC
1000005	6000000	Organisation	CCLRC	CCLRC
1000006	6000000	Organisation	Brunel University	BRU
1000007	6000000	Party	British Broadcasting Corporation	BBC
1000016	6000000	Site	RAL	
1000022	6000000	Site	DL	
1000043	6000000	Organisation	Central Laboratory of the Research Councils	CLRC
1000044	6000000	Department	Business and Information Technology	BITD
1000045	6000000	Division	BITD Directorate Support	BDS
1000046	6000000	Division	E-Business	EBU
1000047	6000000	Division	E-Information	EIN
1000048	6000000	Division	E-Infrastructure	EIF
1000049	6000000	Group	W3C and Hypermedia	W3C
1000050	6000000	Group	Information Science and Engineering	ISE
1000051	6000000	Group	Information Services	INF
1000052	6000000	Group	Enterprise and Learning Solutions	ELS
1000053	6000000	Group	Corporate Systems	CSY
1000054	6000000	Group	Presentation Services Group	PRE
1000055	6000000	Group	Corporate Desktop Support and Development	CDS
1000056	6000000	Group	Networks and Communications	NET
1000057	6000000	Section	Data Management	EDM
1000058	6000000	Section	Trust	TRU
1000059	6000000	Section	Knowledge Systems and Library	KSL
1000060	6000000	Section	Web and Information Systems	WIS
1000061	6000000	Section	Visualisation	VIS

25 rows selected.

OUTPUT

ARC	ENTID1	ENTDOMID1	ENTID2	ENTDOMID2	LEVELP
1	1000005	6000000	1000044	6000000	1
2	1000005	6000000	1000045	6000000	2
3	1000005	6000000	1000046	6000000	2
4	1000005	6000000	1000052	6000000	3
5	1000005	6000000	1000053	6000000	3
6	1000005	6000000	1000047	6000000	2
7	1000005	6000000	1000048	6000000	2
8	1000005	6000000	1000049	6000000	3
9	1000005	6000000	1000050	6000000	3
10	1000005	6000000	1000057	6000000	4
11	1000005	6000000	1000058	6000000	4
12	1000005	6000000	1000059	6000000	4
13	1000005	6000000	1000060	6000000	4
14	1000005	6000000	1000061	6000000	4
15	1000005	6000000	1000051	6000000	3
16	1000005	6000000	1000054	6000000	3
17	1000005	6000000	1000055	6000000	3
18	1000005	6000000	1000056	6000000	3
19	1000044	6000000	1000045	6000000	1
20	1000044	6000000	1000046	6000000	1
21	1000044	6000000	1000052	6000000	2
22	1000044	6000000	1000053	6000000	2
23	1000044	6000000	1000047	6000000	1
24	1000044	6000000	1000048	6000000	1
25	1000044	6000000	1000049	6000000	2
26	1000044	6000000	1000050	6000000	2
27	1000044	6000000	1000057	6000000	3
28	1000044	6000000	1000058	6000000	3
29	1000044	6000000	1000059	6000000	3
30	1000044	6000000	1000060	6000000	3
31	1000044	6000000	1000061	6000000	3
32	1000044	6000000	1000051	6000000	2
33	1000044	6000000	1000054	6000000	2
34	1000044	6000000	1000055	6000000	2
35	1000044	6000000	1000056	6000000	2
36	1000046	6000000	1000052	6000000	1
37	1000046	6000000	1000053	6000000	1
38	1000048	6000000	1000049	6000000	1
39	1000048	6000000	1000050	6000000	1
40	1000048	6000000	1000057	6000000	2
41	1000048	6000000	1000058	6000000	2
42	1000048	6000000	1000059	6000000	2
43	1000048	6000000	1000060	6000000	2
44	1000048	6000000	1000061	6000000	2
45	1000048	6000000	1000051	6000000	1
46	1000048	6000000	1000054	6000000	1
47	1000048	6000000	1000055	6000000	1
48	1000048	6000000	1000056	6000000	1
49	1000050	6000000	1000057	6000000	1
50	1000050	6000000	1000058	6000000	1
51	1000050	6000000	1000059	6000000	1
52	1000050	6000000	1000060	6000000	1
53	1000050	6000000	1000061	6000000	1

53 rows selected.

Network Hierarchy support: REFLEXIVE EXPLODE

FUNCTION cfREFLEXIVE_EXPLODE - the function returns a table of values:

```
SELECT *
FROM TABLE(cfREFLEXIVE_EXPLODE('erContext_Context',
                               'CntxEntId1', 'CntxEntDomId1',
                               'CntxEntId2', 'CntxEntDomId2')) REXCC;
```

PROCEDURE cpREFLEXIVE_EXPLODE – to see the results of the procedure call, query the ciREFLEXIVE_EXPLODE table:

```
BEGIN
 cpREFLEXIVE_EXPLODE('erContext_Context',
                     'CntxEntId1', 'CntxEntDomId1',
                     'CntxEntId2', 'CntxEntDomId2');
END;
/
SELECT * FROM ciREFLEXIVE_EXPLODE;
```

Network Hierarchy support: REFLEXIVE EXPLODE

INPUT

ENTID	ENTDOMID	ENTROLENAME	CNTXNAME	CNTXSHORTNAME
1000004	6000000	Organisation	CLRC	CLRC
1000005	6000000	Organisation	CCLRC	CCLRC
1000006	6000000	Organisation	Brunel University	BRU
1000007	6000000	Party	British Broadcasting Corporation	BBC
1000016	6000000	Site	RAL	
1000022	6000000	Site	DL	
1000043	6000000	Organisation	Central Laboratory of the Research Councils	CLRC
1000044	6000000	Department	Business and Information Technology	BITD
1000045	6000000	Division	BITD Directorate Support	BDS
1000046	6000000	Division	E-Business	EBU
1000047	6000000	Division	E-Information	EIN
1000048	6000000	Division	E-Infrastructure	EIF
1000049	6000000	Group	W3C and Hypermedia	W3C
1000050	6000000	Group	Information Science and Engineering	ISE
1000051	6000000	Group	Information Services	INF
1000052	6000000	Group	Enterprise and Learning Solutions	ELS
1000053	6000000	Group	Corporate Systems	CSY
1000054	6000000	Group	Presentation Services Group	PRE
1000055	6000000	Group	Corporate Desktop Support and Development	CDS
1000056	6000000	Group	Networks and Communications	NET
1000057	6000000	Section	Data Management	EDM
1000058	6000000	Section	Trust	TRU
1000059	6000000	Section	Knowledge Systems and Library	KSL
1000060	6000000	Section	Web and Information Systems	WIS
1000061	6000000	Section	Visualisation	VIS

25 rows selected.

OUTPUT

ARC	ENTID1	ENTDOMID1	ENTID2	ENTDOMID2	LEVELP
1	1000005	6000000	1000005	6000000	0
2	1000005	6000000	1000044	6000000	1
3	1000005	6000000	1000045	6000000	2
4	1000005	6000000	1000046	6000000	2
5	1000005	6000000	1000052	6000000	3
6	1000005	6000000	1000053	6000000	3
7	1000005	6000000	1000047	6000000	2
8	1000005	6000000	1000048	6000000	2
9	1000005	6000000	1000049	6000000	3
10	1000005	6000000	1000050	6000000	3
11	1000005	6000000	1000057	6000000	4
12	1000005	6000000	1000058	6000000	4
13	1000005	6000000	1000059	6000000	4
14	1000005	6000000	1000060	6000000	4
15	1000005	6000000	1000061	6000000	4
16	1000005	6000000	1000051	6000000	3
17	1000005	6000000	1000054	6000000	3
18	1000005	6000000	1000055	6000000	3
19	1000005	6000000	1000056	6000000	3
20	1000044	6000000	1000044	6000000	0
21	1000044	6000000	1000045	6000000	1
22	1000044	6000000	1000046	6000000	1
23	1000044	6000000	1000052	6000000	2
24	1000044	6000000	1000053	6000000	2
25	1000044	6000000	1000047	6000000	1
26	1000044	6000000	1000048	6000000	1
27	1000044	6000000	1000049	6000000	2
28	1000044	6000000	1000050	6000000	2
29	1000044	6000000	1000057	6000000	3
30	1000044	6000000	1000058	6000000	3
31	1000044	6000000	1000059	6000000	3
32	1000044	6000000	1000060	6000000	3
33	1000044	6000000	1000061	6000000	3
34	1000044	6000000	1000051	6000000	2
35	1000044	6000000	1000054	6000000	2
36	1000044	6000000	1000055	6000000	2
37	1000044	6000000	1000056	6000000	2
38	1000045	6000000	1000045	6000000	0
39	1000046	6000000	1000046	6000000	0
40	1000046	6000000	1000052	6000000	1
41	1000046	6000000	1000053	6000000	1
42	1000047	6000000	1000047	6000000	0
43	1000048	6000000	1000048	6000000	0
44	1000048	6000000	1000049	6000000	1
45	1000048	6000000	1000050	6000000	1
46	1000048	6000000	1000057	6000000	2
47	1000048	6000000	1000058	6000000	2
48	1000048	6000000	1000059	6000000	2
49	1000048	6000000	1000060	6000000	2
50	1000048	6000000	1000061	6000000	2
51	1000048	6000000	1000051	6000000	1
52	1000048	6000000	1000054	6000000	1
53	1000048	6000000	1000055	6000000	1
54	1000048	6000000	1000056	6000000	1
55	1000049	6000000	1000049	6000000	0
56	1000050	6000000	1000050	6000000	0
57	1000050	6000000	1000057	6000000	1
58	1000050	6000000	1000058	6000000	1
59	1000050	6000000	1000059	6000000	1
60	1000050	6000000	1000060	6000000	1
61	1000050	6000000	1000061	6000000	1
62	1000051	6000000	1000051	6000000	0
63	1000052	6000000	1000052	6000000	0
64	1000053	6000000	1000053	6000000	0
65	1000054	6000000	1000054	6000000	0
66	1000055	6000000	1000055	6000000	0
67	1000056	6000000	1000056	6000000	0
68	1000057	6000000	1000057	6000000	0
69	1000058	6000000	1000058	6000000	0
70	1000059	6000000	1000059	6000000	0
71	1000060	6000000	1000060	6000000	0
72	1000061	6000000	1000061	6000000	0

72 rows selected.

Network Hierarchy support: TRANSITIVE CLOSURE

FUNCTION cfTRANSITIVE_CLOSURE - the function returns a table of values:

```
SELECT *
FROM TABLE(cfTRANSITIVE_CLOSURE('erContext_Context',
                                'CntxEntId1', 'CntxEntDomId1',
                                'CntxEntId2', 'CntxEntDomId2')) TCCC;
```

PROCEDURE cpTRANSITIVE_CLOSURE – to see the results of the procedure call, query the ciTRANSITIVE_CLOSURE table:

```
BEGIN
 cpTRANSITIVE_CLOSURE ('erContext_Context',
                       'CntxEntId1', 'CntxEntDomId1',
                       'CntxEntId2', 'CntxEntDomId2');
END;
/
SELECT * FROM ciTRANSITIVE_CLOSURE;
```

Network Hierarchy support: TRANSITIVE CLOSURE

INPUT

ENTID	ENTDOMID	ENTROLENAME	CNTXNAME	CNTXSHORTNAME
1000004	6000000	Organisation	CLRC	CLRC
1000005	6000000	Organisation	CCLRC	CCLRC
1000006	6000000	Organisation	Brunel University	BRU
1000007	6000000	Party	British Broadcasting Corporation	BBC
1000016	6000000	Site	RAL	
1000022	6000000	Site	DL	
1000043	6000000	Organisation	Central Laboratory of the Research Councils	CLRC
1000044	6000000	Department	Business and Information Technology	BITD
1000045	6000000	Division	BITD Directorate Support	BDS
1000046	6000000	Division	E-Business	EBU
1000047	6000000	Division	E-Information	EIN
1000048	6000000	Division	E-Infrastructure	EIF
1000049	6000000	Group	W3C and Hypermedia	W3C
1000050	6000000	Group	Information Science and Engineering	ISE
1000051	6000000	Group	Information Services	INF
1000052	6000000	Group	Enterprise and Learning Solutions	ELS
1000053	6000000	Group	Corporate Systems	CSY
1000054	6000000	Group	Presentation Services Group	PRE
1000055	6000000	Group	Corporate Desktop Support and Development	CDS
1000056	6000000	Group	Networks and Communications	NET
1000057	6000000	Section	Data Management	EDM
1000058	6000000	Section	Trust	TRU
1000059	6000000	Section	Knowledge Systems and Library	KSL
1000060	6000000	Section	Web and Information Systems	WIS
1000061	6000000	Section	Visualisation	VIS

25 rows selected.

OUTPUT

ENTID1	ENTDOMID1	ENTID2	ENTDOMID2
1000005	6000000	1000044	6000000
1000005	6000000	1000045	6000000
1000005	6000000	1000046	6000000
1000005	6000000	1000047	6000000
1000005	6000000	1000048	6000000
1000005	6000000	1000049	6000000
1000005	6000000	1000050	6000000
1000005	6000000	1000051	6000000
1000005	6000000	1000052	6000000
1000005	6000000	1000053	6000000
1000005	6000000	1000054	6000000
1000005	6000000	1000055	6000000
1000005	6000000	1000056	6000000
1000005	6000000	1000057	6000000
1000005	6000000	1000058	6000000
1000005	6000000	1000059	6000000
1000005	6000000	1000060	6000000
1000005	6000000	1000061	6000000
1000044	6000000	1000045	6000000
1000044	6000000	1000046	6000000
1000044	6000000	1000047	6000000
1000044	6000000	1000048	6000000
1000044	6000000	1000049	6000000
1000044	6000000	1000050	6000000
1000044	6000000	1000051	6000000
1000044	6000000	1000052	6000000
1000044	6000000	1000053	6000000
1000044	6000000	1000054	6000000
1000044	6000000	1000055	6000000
1000044	6000000	1000056	6000000
1000044	6000000	1000057	6000000
1000044	6000000	1000058	6000000
1000044	6000000	1000059	6000000
1000044	6000000	1000060	6000000
1000044	6000000	1000061	6000000
1000046	6000000	1000052	6000000
1000046	6000000	1000053	6000000
1000048	6000000	1000049	6000000
1000048	6000000	1000050	6000000
1000048	6000000	1000051	6000000
1000048	6000000	1000054	6000000
1000048	6000000	1000055	6000000
1000048	6000000	1000056	6000000
1000048	6000000	1000057	6000000
1000048	6000000	1000058	6000000
1000048	6000000	1000059	6000000
1000048	6000000	1000060	6000000
1000048	6000000	1000061	6000000
1000050	6000000	1000057	6000000
1000050	6000000	1000058	6000000
1000050	6000000	1000059	6000000
1000050	6000000	1000060	6000000
1000050	6000000	1000061	6000000

53 rows selected.

Network Hierarchy support: REFLEXIVE TRANSITIVE CLOSURE

FUNCTION cfREFLEXIVE_TRANSITIVE_CLOSURE - the function returns a table of values:

```
SELECT *
FROM TABLE(cfREFLEXIVE_TRANSITIVE_CLOSURE('erContext_Context',
                                          'CntxEntId1', 'CntxEntDomId1',
                                          'CntxEntId2', 'CntxEntDomId2')) RTCCC;
```

PROCEDURE cpREFLEXIVE_TRANSITIVE_CLOSURE – to see the results of the procedure call, query the ciREFLEXIVE_TRANSITIVE_CLOSURE table:

```
BEGIN
 cpREFLEXIVE_TRANSITIVE_CLOSURE ('erContext_Context',
                                 'CntxEntId1', 'CntxEntDomId1',
                                 'CntxEntId2', 'CntxEntDomId2');
END;
/
SELECT * FROM ciREFLEXIVE_TRANSITIVE_CLOSURE;
```

Network Hierarchy support: REFLEXIVE TRANSITIVE CLOSURE

INPUT

ENTID	ENTDOMID	ENTROLENAME	CNTXNAME	CNTXSHORTNAME
1000004	6000000	Organisation	CLRC	CLRC
1000005	6000000	Organisation	CCLRC	CCLRC
1000006	6000000	Organisation	Brunel University	BRU
1000007	6000000	Party	British Broadcasting Corporation	BBC
1000016	6000000	Site	RAL	
1000022	6000000	Site	DL	
1000043	6000000	Organisation	Central Laboratory of the Research Councils	CLRC
1000044	6000000	Department	Business and Information Technology	BITD
1000045	6000000	Division	BITD Directorate Support	BDS
1000046	6000000	Division	E-Business	EBU
1000047	6000000	Division	E-Information	EIN
1000048	6000000	Division	E-Infrastructure	EIF
1000049	6000000	Group	W3C and Hypermedia	W3C
1000050	6000000	Group	Information Science and Engineering	ISE
1000051	6000000	Group	Information Services	INF
1000052	6000000	Group	Enterprise and Learning Solutions	ELS
1000053	6000000	Group	Corporate Systems	CSY
1000054	6000000	Group	Presentation Services Group	PRE
1000055	6000000	Group	Corporate Desktop Support and Development	CDS
1000056	6000000	Group	Networks and Communications	NET
1000057	6000000	Section	Data Management	EDM
1000058	6000000	Section	Trust	TRU
1000059	6000000	Section	Knowledge Systems and Library	KSL
1000060	6000000	Section	Web and Information Systems	WIS
1000061	6000000	Section	Visualisation	VIS

25 rows selected.

OUTPUT

ENTID1	ENTDOMID1	ENTID2	ENTDOMID2
1000005	6000000	1000005	6000000
1000005	6000000	1000044	6000000
1000005	6000000	1000045	6000000
1000005	6000000	1000046	6000000
1000005	6000000	1000047	6000000
1000005	6000000	1000048	6000000
1000005	6000000	1000049	6000000
1000005	6000000	1000050	6000000
1000005	6000000	1000051	6000000
1000005	6000000	1000052	6000000
1000005	6000000	1000053	6000000
1000005	6000000	1000054	6000000
1000005	6000000	1000055	6000000
1000005	6000000	1000056	6000000
1000005	6000000	1000057	6000000
1000005	6000000	1000058	6000000
1000005	6000000	1000059	6000000
1000005	6000000	1000060	6000000
1000005	6000000	1000061	6000000
1000044	6000000	1000044	6000000
1000044	6000000	1000045	6000000
1000044	6000000	1000046	6000000
1000044	6000000	1000047	6000000
1000044	6000000	1000048	6000000
1000044	6000000	1000049	6000000
1000044	6000000	1000050	6000000
1000044	6000000	1000051	6000000
1000044	6000000	1000052	6000000
1000044	6000000	1000053	6000000
1000044	6000000	1000054	6000000
1000044	6000000	1000055	6000000
1000044	6000000	1000056	6000000
1000044	6000000	1000057	6000000
1000044	6000000	1000058	6000000
1000044	6000000	1000059	6000000
1000044	6000000	1000060	6000000
1000044	6000000	1000061	6000000
1000045	6000000	1000045	6000000
1000046	6000000	1000046	6000000
1000046	6000000	1000052	6000000
1000046	6000000	1000053	6000000
1000047	6000000	1000047	6000000
1000048	6000000	1000048	6000000
1000048	6000000	1000049	6000000
1000048	6000000	1000050	6000000
1000048	6000000	1000051	6000000
1000048	6000000	1000054	6000000
1000048	6000000	1000055	6000000
1000048	6000000	1000056	6000000
1000048	6000000	1000057	6000000
1000048	6000000	1000058	6000000
1000048	6000000	1000059	6000000
1000048	6000000	1000060	6000000
1000048	6000000	1000061	6000000
1000049	6000000	1000049	6000000
1000050	6000000	1000050	6000000
1000050	6000000	1000057	6000000
1000050	6000000	1000058	6000000
1000050	6000000	1000059	6000000
1000050	6000000	1000060	6000000
1000050	6000000	1000061	6000000
1000051	6000000	1000051	6000000
1000052	6000000	1000052	6000000
1000053	6000000	1000053	6000000
1000054	6000000	1000054	6000000
1000055	6000000	1000055	6000000
1000056	6000000	1000056	6000000
1000057	6000000	1000057	6000000
1000058	6000000	1000058	6000000
1000059	6000000	1000059	6000000
1000060	6000000	1000060	6000000
1000061	6000000	1000061	6000000

72 rows selected.

Valid Time support

- Called *Entity Time* at the CC level
- *Entity Time* is represented by the following timepoints:
  - Begin
  - End
- *Entity Time Type* indicates the time type.
- The CDR implementation of *Valid Time* is based on the book "*Temporal Data And The Relational Model*" by Date, Darwen and Lorentzos (2003).
- *Valid Time* in the CDR is associated with an entire row of data in a table.
- The meaning of *Valid Time* for a given Entity Class must never change.
- The meaning of *Valid Time* may differ for each Entity Class.

CCLRC

Valid Time support

The CDR currently supports the following ***Valid Time*** procedures and functions:

Functions:

cfBeginningOfTime – return the beginning of SQL time is 01-Jan-0001 00:00:00
cfEndOfTime – return the end of SQL time is 31-Dec-9999 23:59:59
cfTimeQuantum – return the smallest time unit of time in the current CDR is set to 1 second
cfVT_AFTER - does i1 occur after i2?
cfVT_BEFORE - does i1 occur before i2?
cfVT_BEGINS - does i1 begin at the same point as i2?
cfVT_BETWEEN - does t3 occur between t1 and t2?
cfVT_CURRENT - does the CURRENT_TIMESTAMP occur between t1 and t2?
cfVT_ENDS - does i1 end at the same point as i2?
cfVT_EQUALS - are i1 and i2 equal?
cfVT_INCLUDED_IN – is i1 included in i2?
cfVT_INCLUDES - does i1 include i2?
cfVT_INTERSECT - return the intersection of i1 with i2
cfVT_MAX - return the greater time; t1 or t2
cfVT_MEETS - does i1 meet i2 to a resolution of cfTimeQuantum or less?
cfVT_MEETS_DS - does i1 meet i2 to a given resolution of days or seconds?
cfVT_MEETS_YM - does i1 meet i2 to a given resolution of years or months?
cfVT_MERGES - does i1 merge with i2? (i.e. does i1 MEET or OVERLAP i2)?
cfVT_OVERLAPS - does i1 overlap with i2?
cfVT_UNION - return the union of i1 with i2

Procedures:

cpVT_INTERSECT - return the intersection of i1 with i2
cpVT_MINUS - return the difference between i1 with i2
cpVT_UNION - return the union of i1 with i2

Entity Time Type

The clEntity attribute EntTimeType stores the type of the Entity time.

Here are the time types:

0 Current Time or 'Now'

1000 Time Begin Point
1001 Time End Point

2000 Time Period [Begin closed, End closed]
2001 Time Period [Begin closed, End open)
2002 Time Period (Begin open, End closed]
2003 Time Period (Begin open, End open)

3000 Time Interval [Begin closed, End closed]
3001 Time Interval [Begin closed, End open)
3002 Time Interval (Begin open, End closed]
3003 Time Interval (Begin open, End open)

A closed point in a period or interval includes that point and is shown textually by a bracket [or].
An open point in a period or interval excludes that point and is shown textually by a parenthesis (or).

NOTE
1. A time Period is anchored on at least one end to the timeline. A time Interval is not anchored.
2. A time point value of NULL means that that time value is unknown.
3. The earliest possible ISO SQL time is 01-Jan-0001 00:00:00' and is known as the BeginningOfTime.
4. The latest possible ISO SQL time is '31-Dec-9999 23:59:59' and is known as the EndOfTime.
5. Use the BeginningOfTime and EndOfTime to create a practically infinite time axis suitable for most applications.

CCLRC

Entity Time Type

Examples of Time Types

EXAMPLE 1:
Assuming an ordering from Sunday to Saturday, all the following examples are equivalent and include only the days: Mon, Tue, Wed, Thu, Fri:

1. [Mon, Fri]
2. [Mon, Sat)
3. (Sun, Fri]
4. (Sun, Sat)

EXAMPLE 2:
The following period extends from 1 Apr 2004 (inclusive) to 31 Mar 2005 (inclusive):
[1-Apr-2004, 31-Mar-2005]

EXAMPLE 3:
The following period extends from an unknown time in the past to the 24 Nov 2005:
[NULL, 24-Nov-2005]

EXAMPLE 4:
The following example extends from 31 Mar 2005 until an unknown time in the future:
[31-Mar-2005, NULL]

EXAMPLE 5:
The following example creates a time period that effectively spans the whole of known time for most applications:
[cfBeginningOfTime, cfEndOfTime]

EXAMPLE 6:
The following example creates a time period that extends from an unknown time in the past to an unknown time in the future:
[NULL, NULL]

Valid Time Time

Valid Time - *the period over which a fact was believed to be true in the real world.*

Our *beliefs* may change - so may valid time.

Example: *King Canute lived from 995 to 1035.*

erPerson

ObjId	ObjDomId	EntId	EntDomId	ValTimeBegin	ValTimeEnd	PerFamilyNames	PerFirstNames
1000100	9000000	1000081	9000000	01 Jan 995 00:00	12 Nov 1035 00:00	Canute	King
1000101	9000000	1000083	9000000	01 Apr 1995 09:00	18 Sep 2002 09:00	Smith	Mary
1007000	9000000	1000083	9000000	19 Sep 2002 09:00	NULL	Jones	Mary
1000107	9000000	1000089	9000000	01 Apr 1995 09:00	NULL	Baker	Benjamin John
1000099	6000000	1000089	9000000	01 Apr 1995 15:00	NULL	Baker	Benjamin

Transaction Time support

- Called *Object Time* at the CC level.

- *Object Time* is represented by the following timepoints:
  - Insert - when the row was created
  - Update - when the row was updated
  - Delete - when the row was logically deleted

- *Object Time Type* indicates the time type.

- *Transaction Time* in the CDR applies to the *entire* row of attributes in a table, *including* the *Valid Time* attributes.

Transaction Time

Transaction Time - *the period over which a fact was recorded to exist in the database.*

Our *records* do not change – neither does transaction time.

Example: The fact "*King Canute lived from 995 to 1035*" was recorded to exist in the database from 7 *Mar 1995* until *now*.

ccPerson

ObjId	ObjDomId	ObjTimeInsert	ObjTimeUpdate	ObjTimeDelete	EntId	EntDomId	PerFamilyNames	PerFirstNames
1000100	9000000	07 Mar 1995 09:00	NULL	NULL	1000081	9000000	Canute	King
1000101	9000000	20 Mar 1995 09:00	NULL	NULL	1000083	9000000	Smith	Mary
1007000	9000000	19 Sep 2002 09:00	NULL	NULL	1000083	9000000	Jones	Mary
1000107	9000000	01 Apr 1995 09:00	04 Apr 1995 11:00	NULL	1000089	9000000	Baker	Benjamin John
1000099	6000000	01 Apr 1995 15:00	02 Apr 1995 10:00	NULL	1000089	9000000	Baker	Benjamin
1000101	6000000	01 Apr 2005 10:00	NULL	02 Apr 2005 12:00	1000089	6000000	Doe	John

Object Time Type

The clObject attribute ObjTimeType stores the type of the Object time.

Here are the time types:

0 Current Time or 'Now'

1000 Time Begin Point
1001 Time End Point

2000 Time Period [Begin closed, End closed]
2001 Time Period [Begin closed, End open)
2002 Time Period (Begin open, End closed]
2003 Time Period (Begin open, End open)

3000 Time Interval [Begin closed, End closed]
3001 Time Interval [Begin closed, End open)
3002 Time Interval (Begin open, End closed]
3003 Time Interval (Begin open, End open)

A closed point in a period or interval includes that point and is shown textually by a bracket [or].
An open point in a period or interval excludes that point and is shown textually by a parenthesis (or).

NOTE
1. A time point value of NULL means that that time value is unknown.
2. The earliest possible ISO SQL time is 01-Jan-0001 00:00:00' and is known as the BeginningOfTime.
3. The latest possible ISO SQL time is '31-Dec-9999 23:59:59' and is known as the EndOfTime.
4. Use the BeginningOfTime and EndOfTime to create a practically infinite time axis suitable for most applications.

Multiple Language support

- Unicode attributes data types:
  - All character attribute values are encoded in Unicode (AL32UTF8)

- Multiple language rows:
  - Multiple rows store similar information in multiple languages.
  - Each Object or row has a single *ISO 639-2* language code.
  - *ISO 639-2* language codes are stored in the **`clLanguage`** table.
  - All External ER level views have the English language (“eng”) hard-coded into the associated INSERT trigger.

Multiple Entity Signs support

Symbol:

- Name
- Synonym
- Acronym
- Abbreviation
- Nickname
- *various* Identifiers (e.g. Federal, Grid)

Index:

- URL
- Email Address
- Postal Address

Icon:

- Image

ISO Standard SQL3 attribute data types

The ISO SQL3 (1999) standard supports the following attribute data types, which are now implemented in ERwin by linking-in the *Data Types* file: CDR2005_DSM007.dsm.

Only these ISO data types should be chosen for CDR attributes.

ERwin Domain	**ISO SQL3 Data Type**	**ERwin Subdomain**
Blob	BLOB	BLOB
Datetime	INTERVAL DAY TO SECOND	INTERVAL_DAY_TO_SECOND
	INTERVAL YEAR TO MONTH	INTERVAL_YEAR_TO_MONTH
	TIMESTAMP WITH TIME ZONE	TIMESTAMP_WITH_TIME_ZONE
Number	DECIMAL(n)	DECIMAL
	DECIMAL(n, m)	DECIMAL2
	FLOAT	FLOAT
	INTEGER	INTEGER
	NUMERIC(n)	NUMERIC
	NUMERIC(n, m)	NUMERIC2
String	NCHAR	NCHAR1
	NCHAR(n)	NCHAR2, NCHAR4, NCHAR8, NCHAR16, NCHAR32, NCHAR64
	NVARCHAR(n)	NVARCHAR4, NVARCHAR8, NVARCHAR16, NVARCHAR32, NVARCHAR64, NVARCHAR128, NVARCHAR256, NVARCHAR1024
	NCLOB	NCLOB

Glossary of table and attribute names & abbreviations

The *Glossary* is implemented in ERwin by linking-in the *Names* file: CDR2005_NSM007.nsm.

The *Glossary* of names stores the full names and abbreviations necessary to map from the ERwin Logical view to Physical view in the data model e.g. “RutherfordAppletonLaboratory” and “RAL”

The Glossary can be accessed in ERwin menus:

- Tools > Names > Model Naming Options
- Push the “Edit” button
- After editing, save the results: File > Save
- Push the “Reload” button (a reload confirmation message must appear to be sure this has worked)

2. CDR Data Model Changes

We shall examine changes made to the CDR data model since version CDM2003_309.
The current CDR data model is CDR2005_CDM007_145.

These changes include:

1. Entity Id (Logical Id) and Object Id (Physical Id)
2. Entity Domain and Object Domain (Database Id)
3. Object Class Hierarchy a.k.a. "Ontology"
4. Object Classes (Type Id) and Entity Roles (Role Name)
5. Owner Id (Database User Id)
6. Diagram layout and colour conventions
7. ISO Standard SQL3 attribute data types (ERwin standard data types)
8. Valid Time functions and procedures
9. Transaction Time (clTransaction table)
10. Network Hierarchy functions and procedures
11. Multiple Language Rows
12. Object row visibility and Object row order

NOTE: all items in parentheses are references to the old data model

3. CDR Data Modelling Best Practices

1. Table and attribute naming conventions
    1. No renaming of attributes at the CC level, only at EXT level
2. Attribute maintenance
3. Names maintenance
4. Data Type maintenance
5. Code maintenance:
    1. View WHERE clauses
    2. Stored INSTEAD OF triggers
    3. Stored procedures and functions
    4. Fonts and code layout
    5. Do not use TABS
6. Data model layout
7. Colour conventions
8. Managing change:
    1. Change procedure
    2. Release procedure

Code maintenance

Code maintenance:

- View WHERE clauses
- Stored INSTEAD OF triggers
- Stored procedures and functions
- Fonts and code layout
- Do not use TABS, only spaces
- Use Microsoft *Notepad* or equivalent
- Use only *Courier* or *Courier New* font
- Do not mix coding styles and layouts.

Managing change

Change procedure

1. Lock the *Lock* file
2. Copy the data model from the public area to a private area
3. Increment the data model version
4. Make changes to data model using Notepad and a *Courier* font
5. Document the changes in the *Changes* spreadsheet
6. Test the changes in the data model by Forward Generating in ERwin against a Oracle Test database. Check all code for errors using Oracle Enterprise Manager.
7. Complete the *Change Procedure* checklist.

Managing change

Release procedure

1. Copy the new data model, names file and data types file from the private area to the public area. Create a subfolder to hold this release.
2. Double-check the released files are indeed the correct ones
3. Create a *Release Notes* file containing the following information:
    1. Name of this release
    2. Date and time of this release
    3. Person issuing this release
    4. A complete list of files for this release
    5. Location of all files for this release
    6. Additional notes or build instructions
4. Complete the *Release Procedure* checklist.
5. Send an email to the **CDR_Release** group with the *Release Notes* file attached.
6. Unlock the *Lock* file.

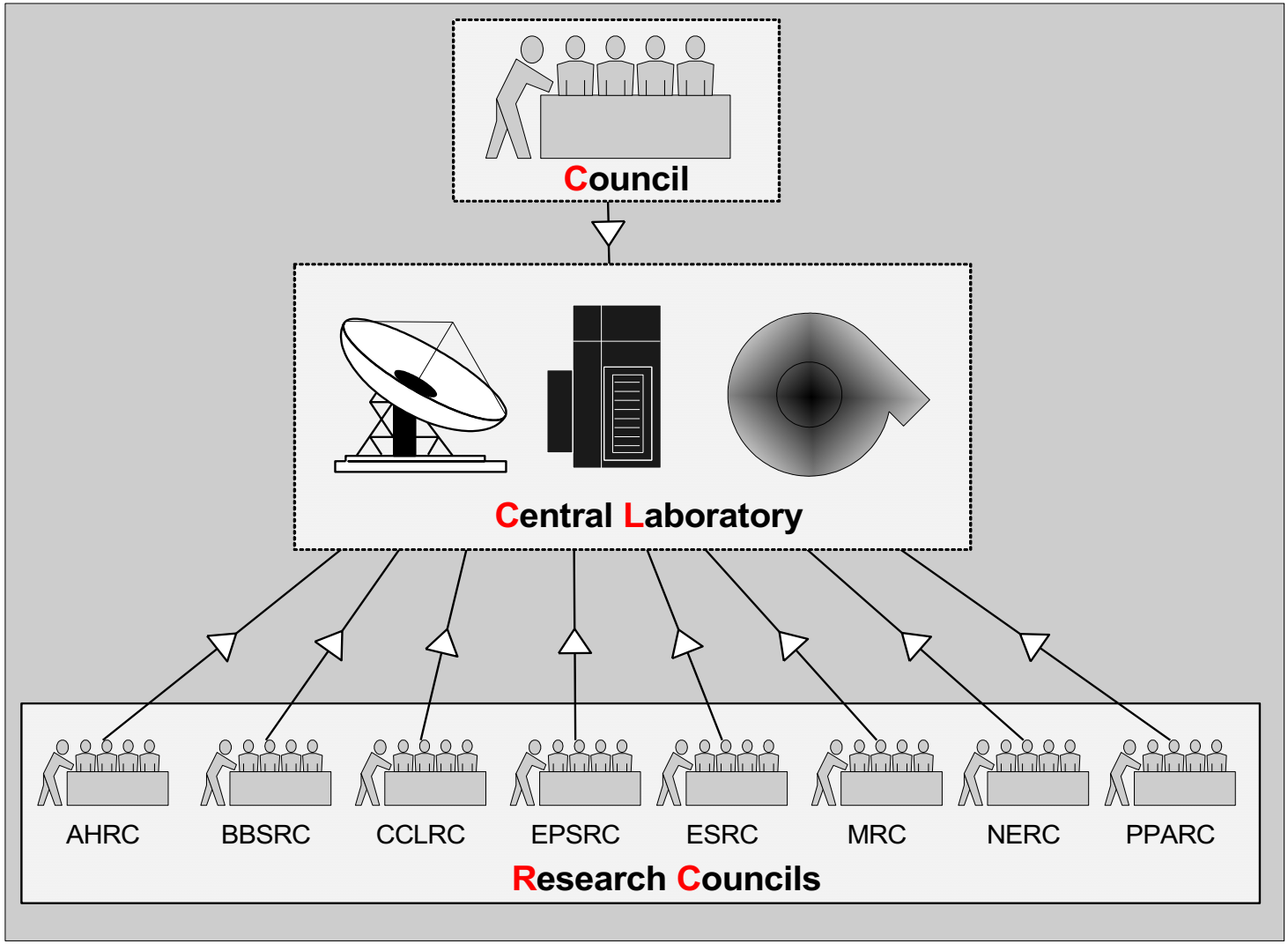
Conclusion

In these three workshops we:

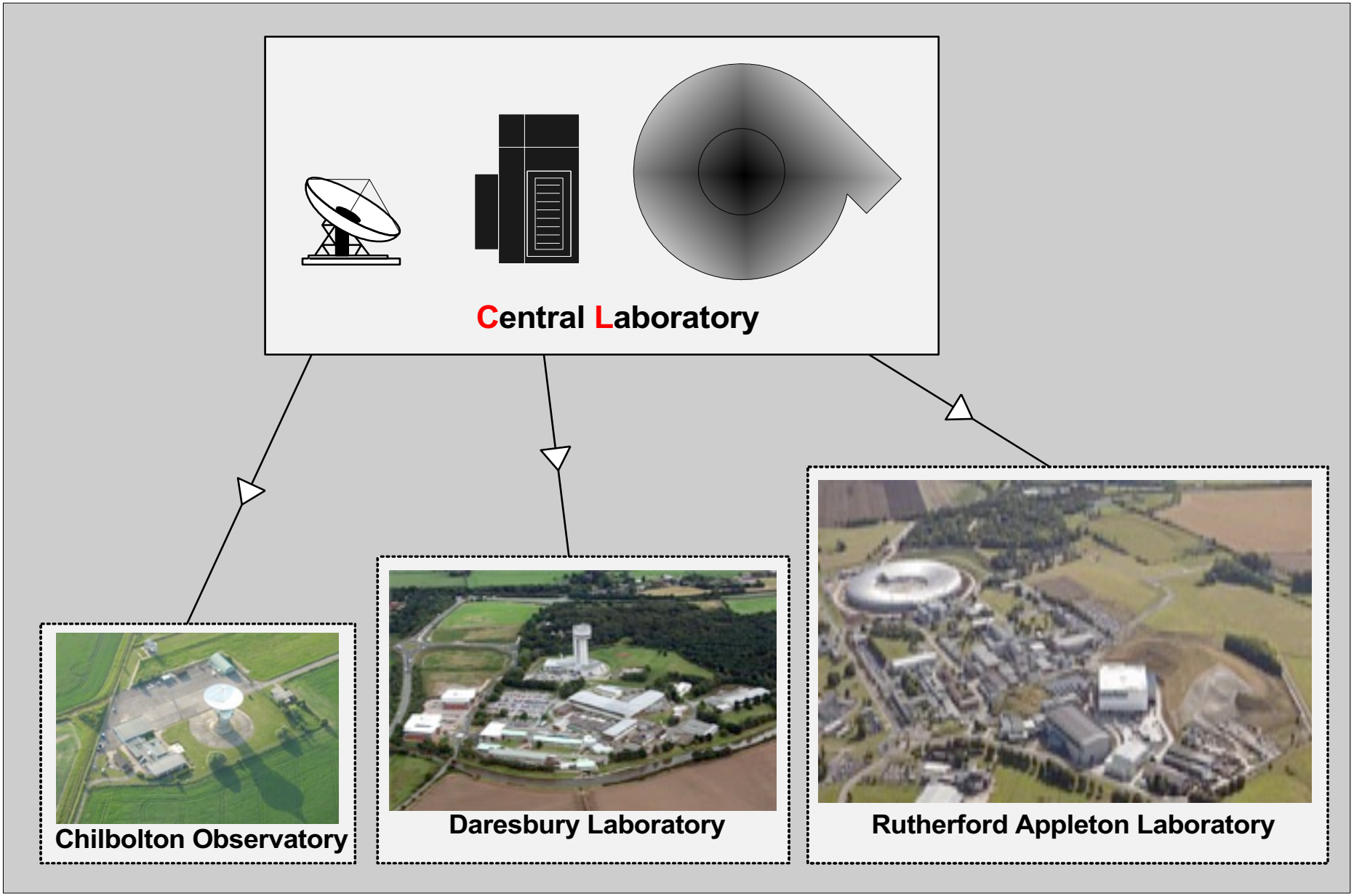
1. *Reviewed* CDR Data Model Features.
2. *Highlighted* CDR Data Model Changes.
3. *Recommended* CDR Data Modelling Best Practices.

Council for the Central Laboratory of the Research Councils

(1995)



Central Laboratory



Chilbolton Observatory

Chilbolton Observatory

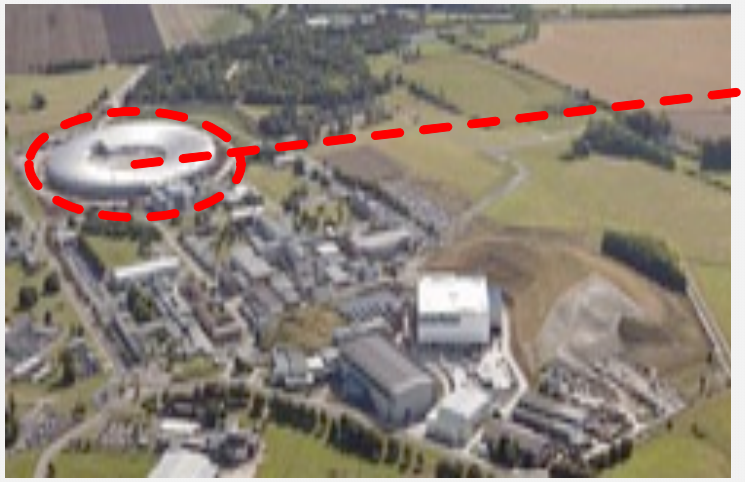
- This site in Hampshire opened in 1967
- *Chilbolton Dish* - 25 metre radar antenna:
  - *CAMRa* Rain Radar
  - *ACROBAT* Clear-air Radar
- *Galileo* Cloud Radar
- *Copernicus* Doppler Cloud Radar
- *Raman* Lidar – measures atmospheric properties
- Various meteorological sensors and several other services

Daresbury Laboratory

Daresbury Laboratory

- This site in Cheshire opened in 1962
- *Synchrotron Radiation Source* (SRS):
  - Creates intense X-ray light
  - Service closes in 2008
- *HPCx* service of 1280 processors:
  - 40 *IBM pSeries 690 Regatta* nodes
  - 32 *POWER4* processors per node
- *eScience* and several other services

Rutherford Appleton Laboratory



Rutherford Appleton Laboratory

Diamond Light Source (DLS)

- Site opened in 2005
- Synchrotron service starts in 2007
- Creates intense X-ray & UV light

- This site in Oxfordshire opened in 1957
- *Atlas* Computing Laboratory (since 1975)
- *Central Laser Facility* (CLF) – high power lasers
- *ISIS* – pulsed neutron and muon sources
- *RAL Ground Station Antenna* – 12 metre radar antenna
- *Space test facility* – imitates the conditions of space
- *Starlink* – processes and analyses astronomical data
- *eScience* and several other services

CCLRC

Corporate Data *Muddle*

The Corporate Data *Muddle*

Symptoms of the data muddle:

- *Cannot use a single security pass across sites*
- *Laptops need to be configured for each site*
- *Cannot book meeting rooms across sites*
- *Cannot book taxis across sites*
- *Different library cards for each site*
- *In short, you feel like a persona non grata*

Dangers of the data muddle:

- *Sharing data across sites can result in multiple records for the same person*

The Corporate Data *Muddle*

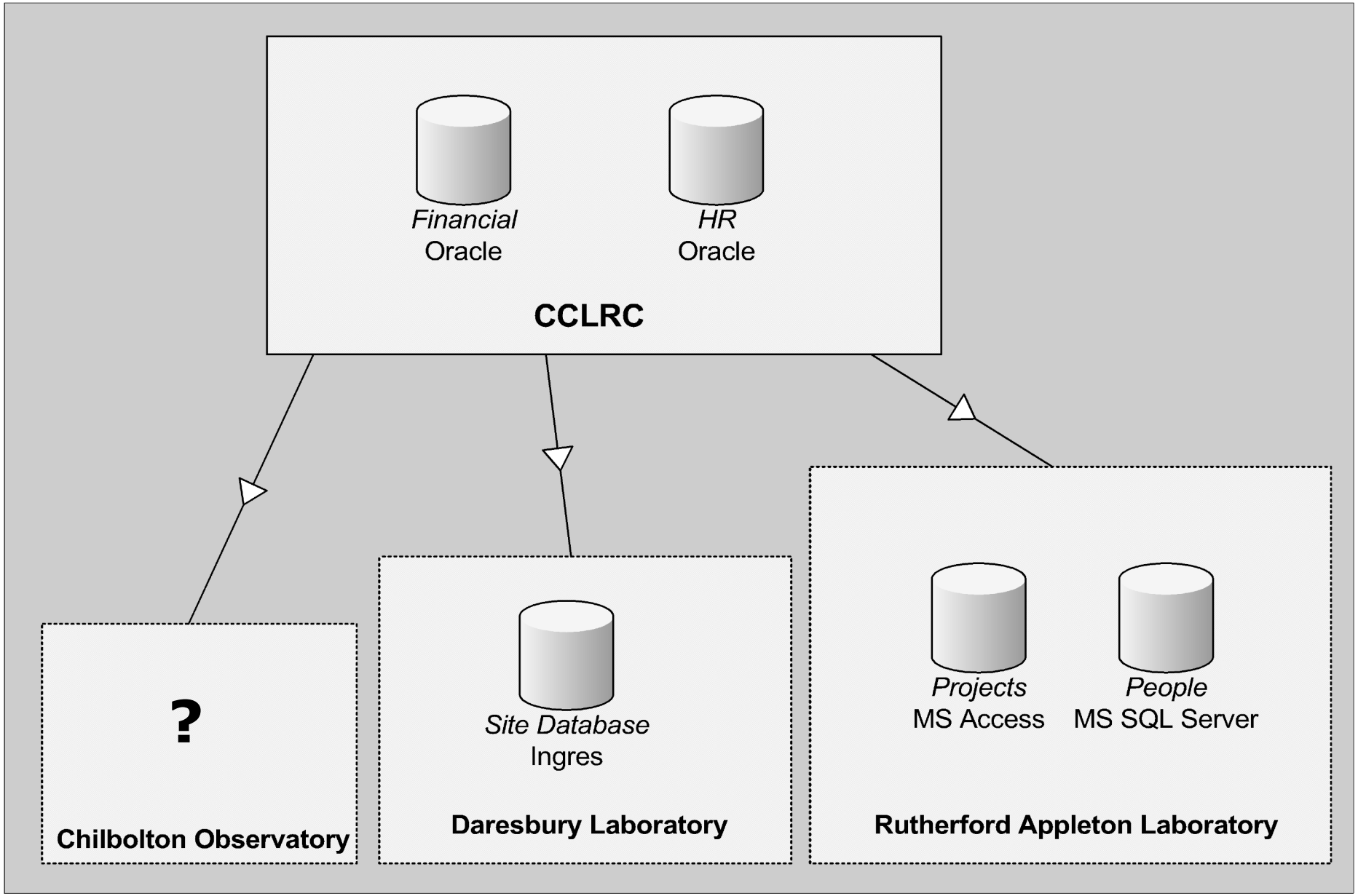
Consequences of the data muddle:

- *Different database vendors*
- *Different data models*
- *Different application technologies*
- *Different services at each site*
- *Incomplete data held for each site*
- *Develop similar applications for each site*
- *Entrenched development philosophies per site*

Causes of the data muddle:

- *Site history*
- *Corporate re-organisations*

The Corporate Data *Muddle*



The Corporate Data *Muddle*

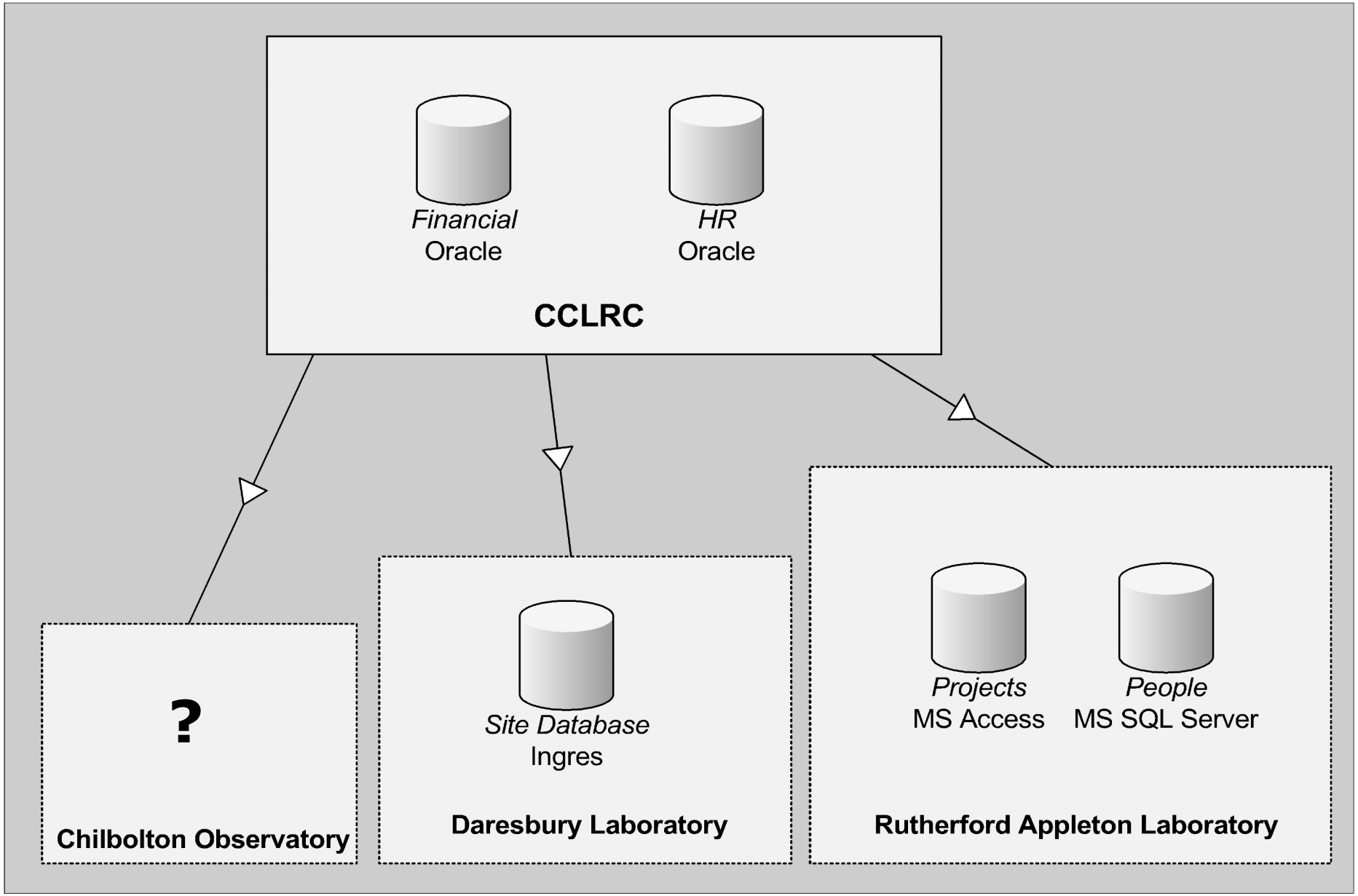
Some architectural solutions:

- A *single* corporate database for *all* applications
  - *Very complex and expensive to develop.*
- A *limited* corporate database feeding all departmental databases
  - *Probably the best compromise.*
- Corporate applications that work transparently across sites
  - *Web based applications using open standards.*

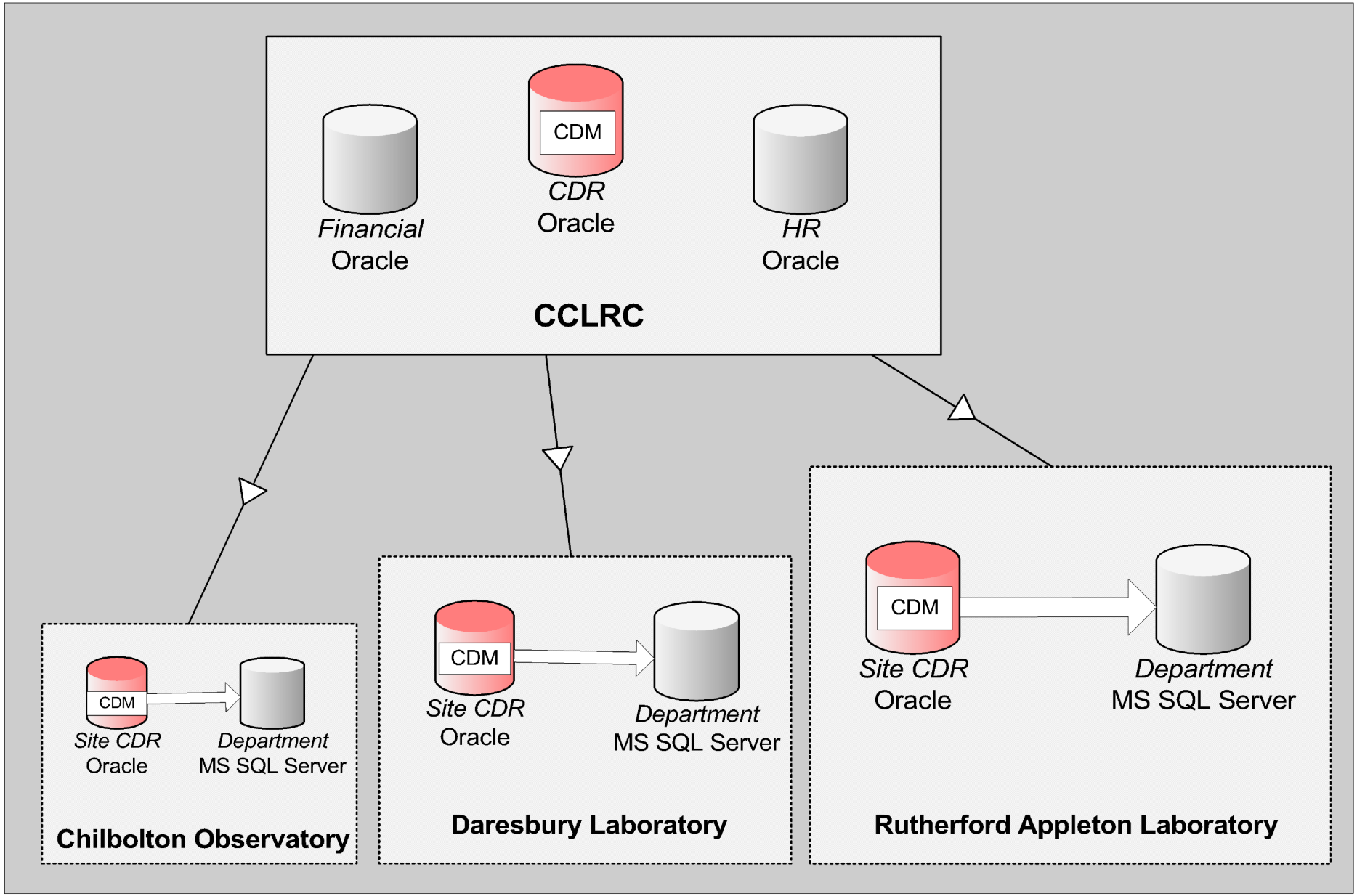
CCLRC

Corporate Data Repository (CDR) Requirements

The Corporate Data Repository



The Corporate Data Repository



The Corporate Data Repository

- Holds *common* corporate data
- Includes information about:
  - Resources: persons, infrastructure
  - Organisation Units: organisations, departments
  - Roles
  - Authorisations
- May include information about:
  - Projects
  - Publications
  - Bookings
- But *not* data on:
  - Company Finances
  - Personal private information

The Corporate Data Repository

- Sometimes a *repository* is defined to be a meta-data layer above the database layer.
- We define a *repository* to mean a *global* database that supports *local* databases.
- Analogous to global and local variables in a computer program.
- The CDR hosts the *Corporate Data Model* (CDM).

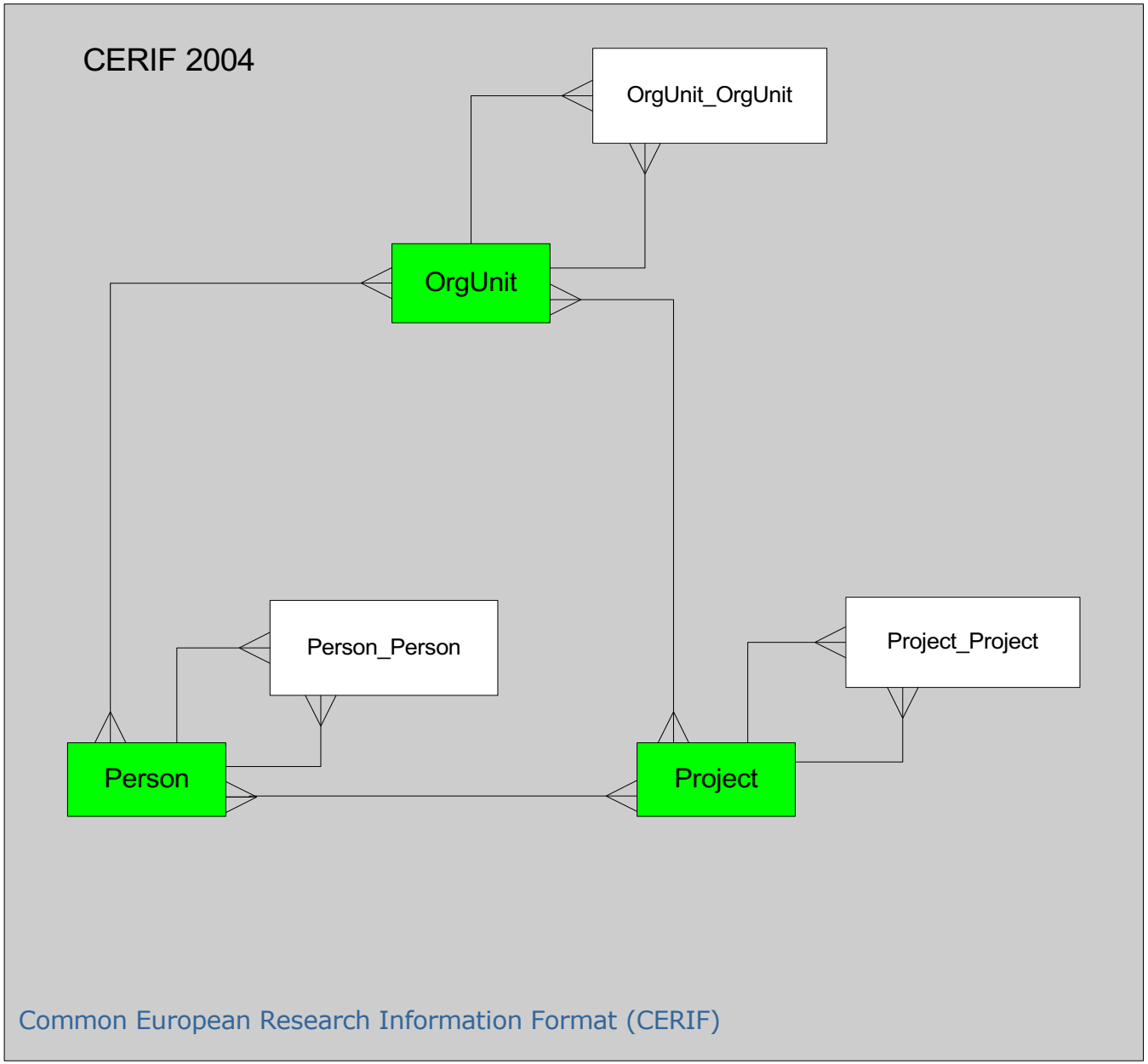
CCLRC

Corporate Data Model (CDM) Design

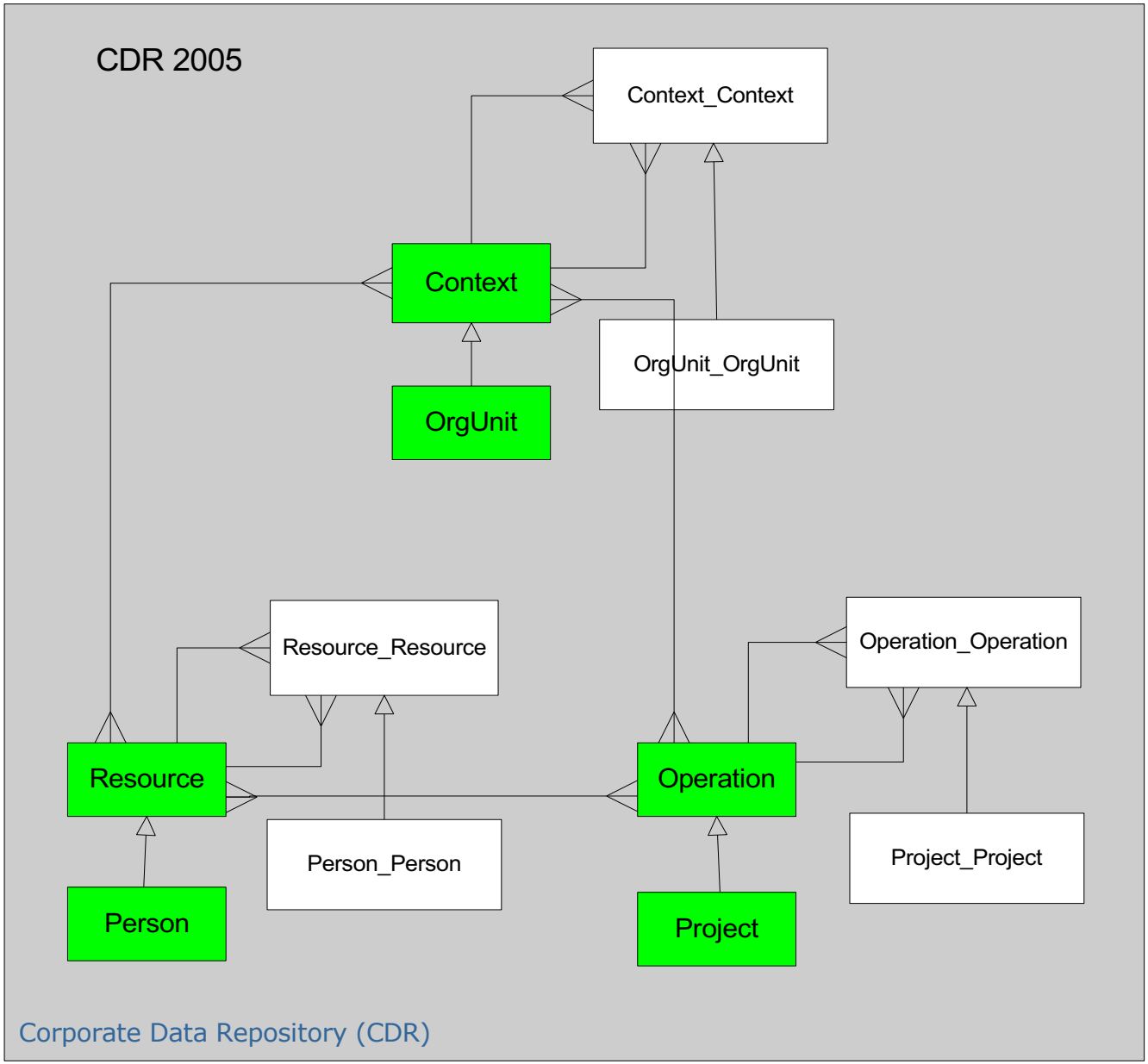
Corporate Data Model (CDM)

- Currently the CDM models:
  - *People* related to CCLRC and DLS:
    - roles
    - authorisations
  - *Organisation Units* in CCLRC and DLS
  - *Projects*
- Consolidates data from the existing site databases at RAL and DL
- Based on the *CERIF 2004* standard:
  - http://www.eurocris.org/en/taskgroups/cerif/

CERIF 2004


CERIF 2004
OrgUnit_OrgUnit
OrgUnit
Person_Person
Project_Project
Person
Project
Common European Research Information Format (CERIF)

CDR 2005


CDR 2005
Context_Context
Context
OrgUnit_OrgUnit
OrgUnit
Resource_Resource
Operation_Operation
Resource
Operation
Person_Person
Project_Project
Person
Project
Corporate Data Repository (CDR)

CDM Design Decisions

- Based on the *CERIF 2004* standard
- Based on *ANSI/SPARC Architecture*
- Supports temporal data at the row level
- Supports data distribution
- Supports globalisation: *Unicode* and time zones
- Supports inheritance for entities and relationships
- Supports complex relationships
- Supports recursive relationships
- Accessible through open standards interfaces: *JDBC*, *ODBC*
- Implementable on more than one database: *Oracle*, *Ingres*, *Microsoft SQL Server*

CDM Entities and Objects

- The *Interim Report* (1975) of the *ANSI/X3/SPARC* Study Group on DBMSs stated the following:

  "*An object in the real world is called an entity*"

- A paper by *Hall, Owlett* and *Todd* (1976) discusses the concept of a *surrogate*:

  "… *every entity of the outside world is associated with a surrogate which stands for that object in the model*"

- A *surrogate* is an object in the model that is in one-to-one correspondence with an entity in the world

CDM Entity and Object Identifiers

Entity Identifier

- an identifier used to identify the unique existence of an *entity* in the real world.
- a *surrogate*.

Object Identifier

- an identifier used to identify the unique existence of an *object* in the database.
- a global *primary key*.

NOTE
An Entity Identifier may have several Object Identifiers.

erPerson

ObjId	ObjDomId	EntId	EntDomId	ValTimeBegin	ValTimeEnd	PerFamilyNames	PerFirstNames
1000100	9000000	1000081	9000000	01 Jan 995 00:00	12 Nov 1035 00:00	Canute	King
1000101	9000000	1000083	9000000	01 Apr 1995 09:00	18 Sep 2002 09:00	Smith	Mary
1000105	9000000	1000083	9000000	19 Sep 2002 09:00	NULL	Jones	Mary
1000107	9000000	1000089	9000000	01 Apr 1995 09:00	NULL	Baker	Benjamin

CDM Entity and Object Domains

Each Identifier belongs to a corresponding Domain:

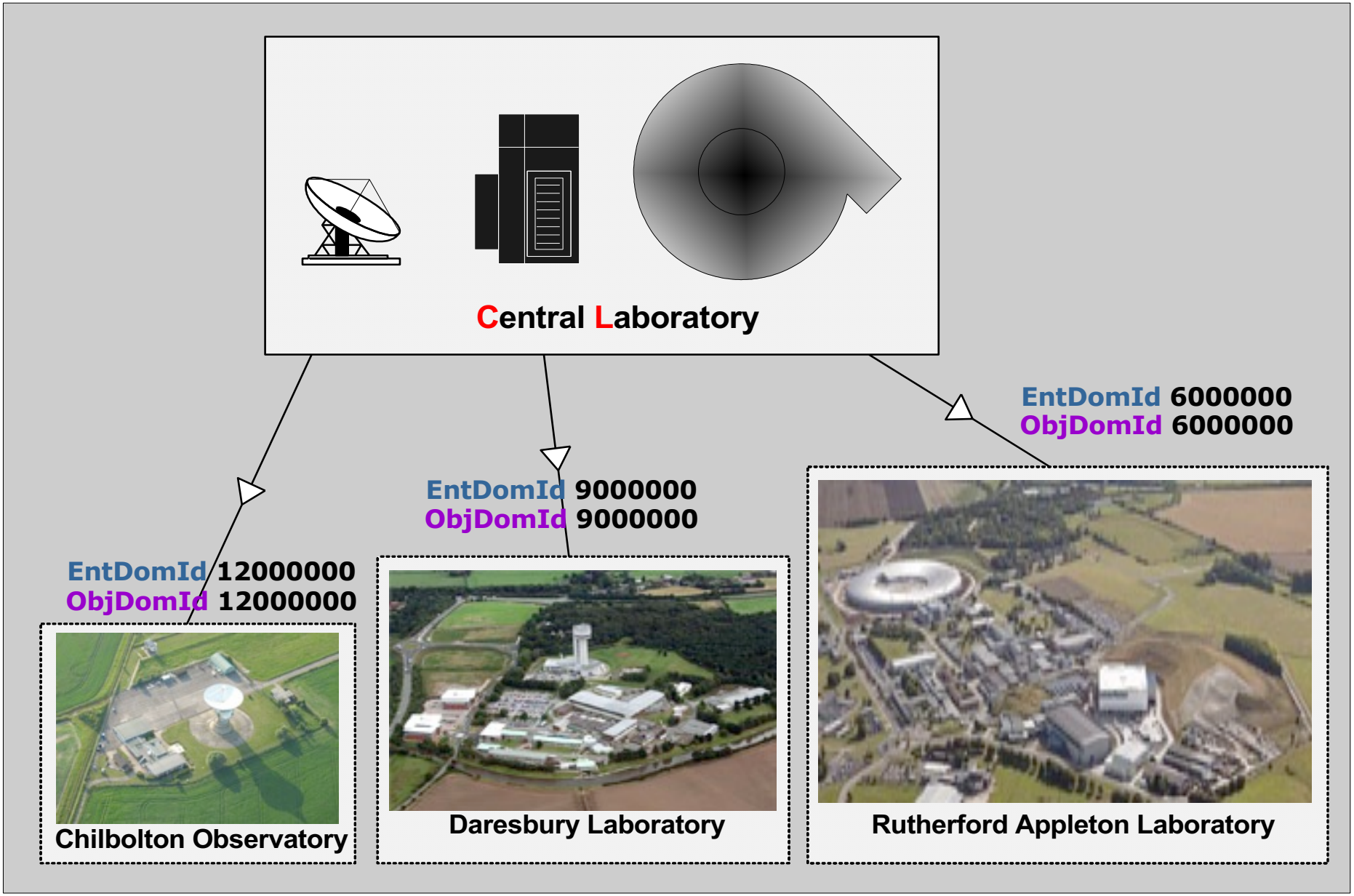
Entity Domain
: the *entity* domain to which an Entity Identifier belongs.

Object Domain
: the *object* domain to which an Object Identifier belongs.

ccPerson

ObjId	ObjDomId	EntId	EntDomId	PerFamilyNames	PerFirstNames
1000100	9000000	1000081	9000000	Canute	King
1000101	9000000	1000083	9000000	Smith	Mary
1000105	9000000	1000083	9000000	Jones	Mary
1000107	9000000	1000089	9000000	Baker	Benjamin John
1000999	6000000	1000089	9000000	Baker	Benjamin

Central Laboratory Domains



CCLRC

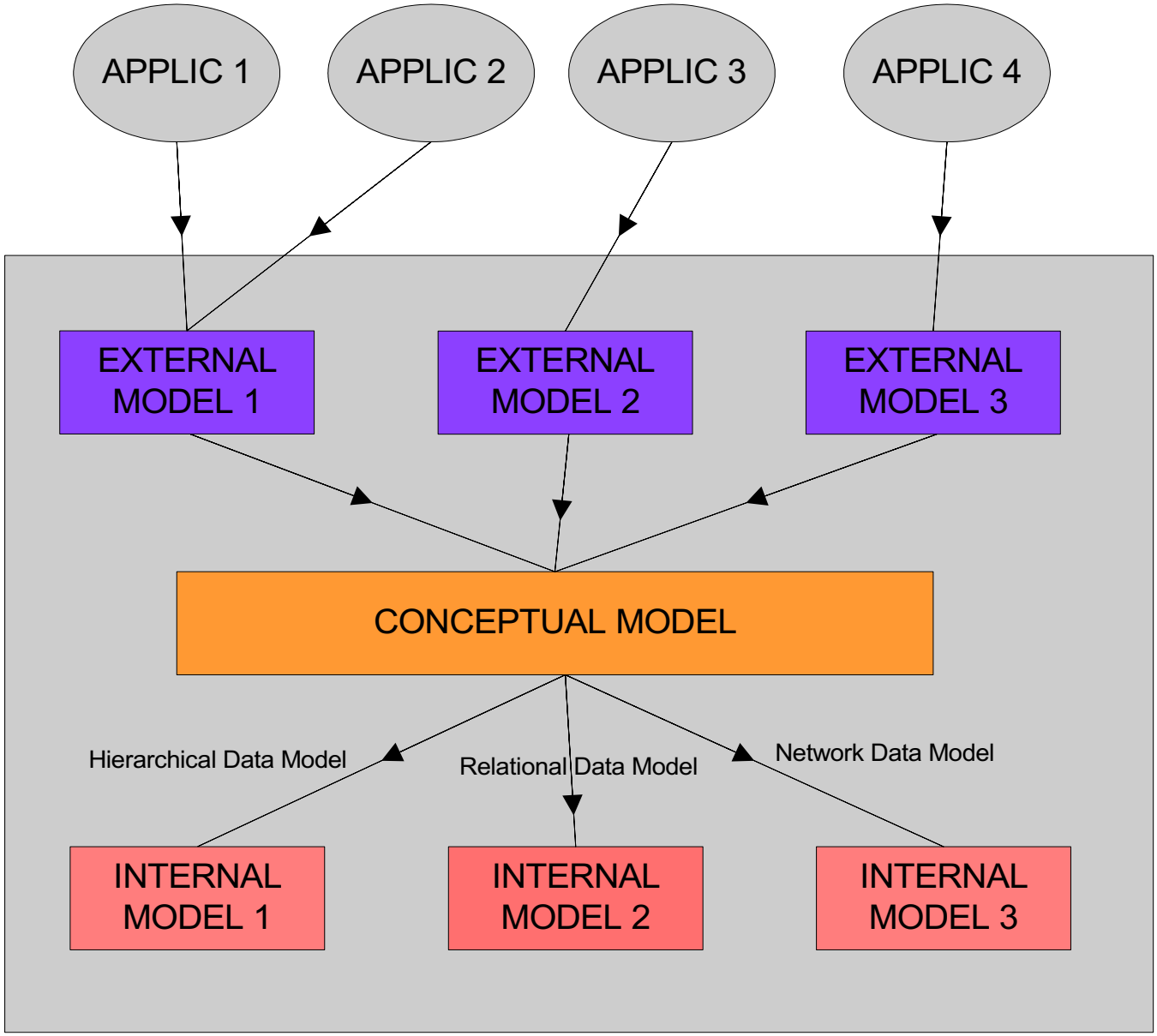
CDM Architecture

ANSI/X3/SPARC Architecture

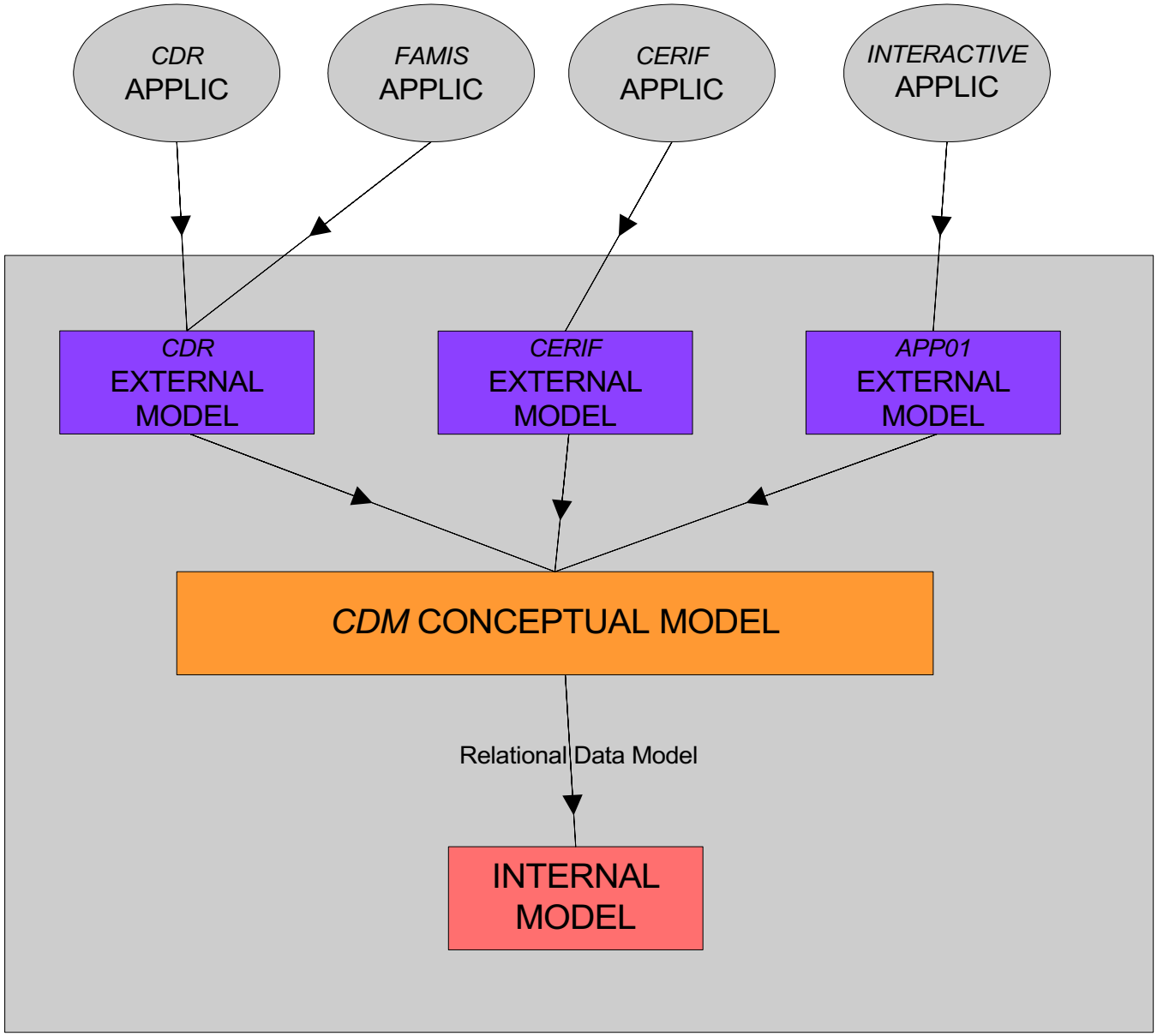
The *Interim Report* (1975) of the *ANSI/X3/SPARC* Study Group on DBMSs also proposed three "realms of interest" in a database architecture:

- External realm: a simplified model of the real world as seen by *one or more* applications.
- Conceptual realm: a limited model of the real world for *all* applications.
- Internal realm: data in computer storage representing the *conceptual realm*.

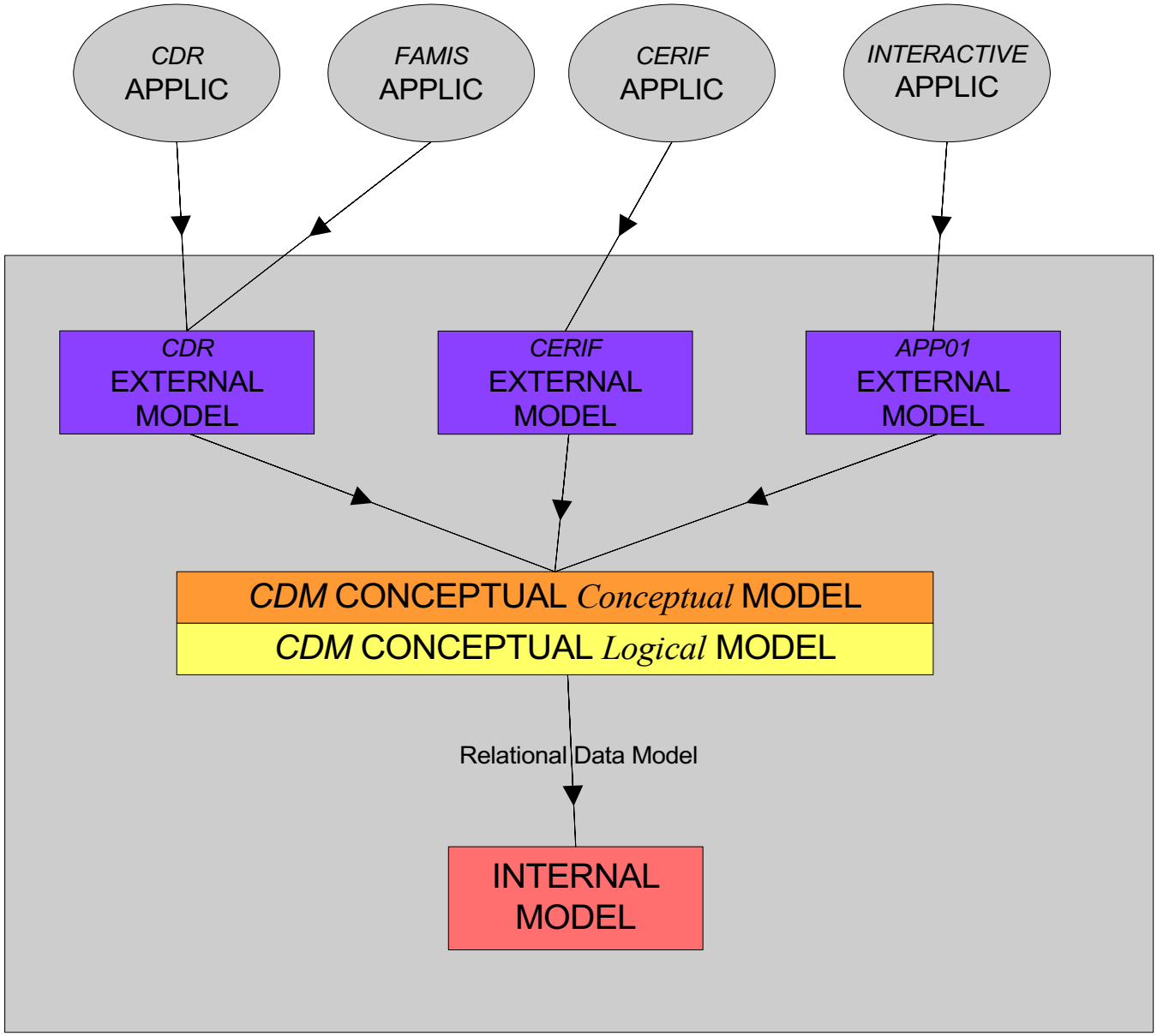
ANSI/X3/SPARC Architecture



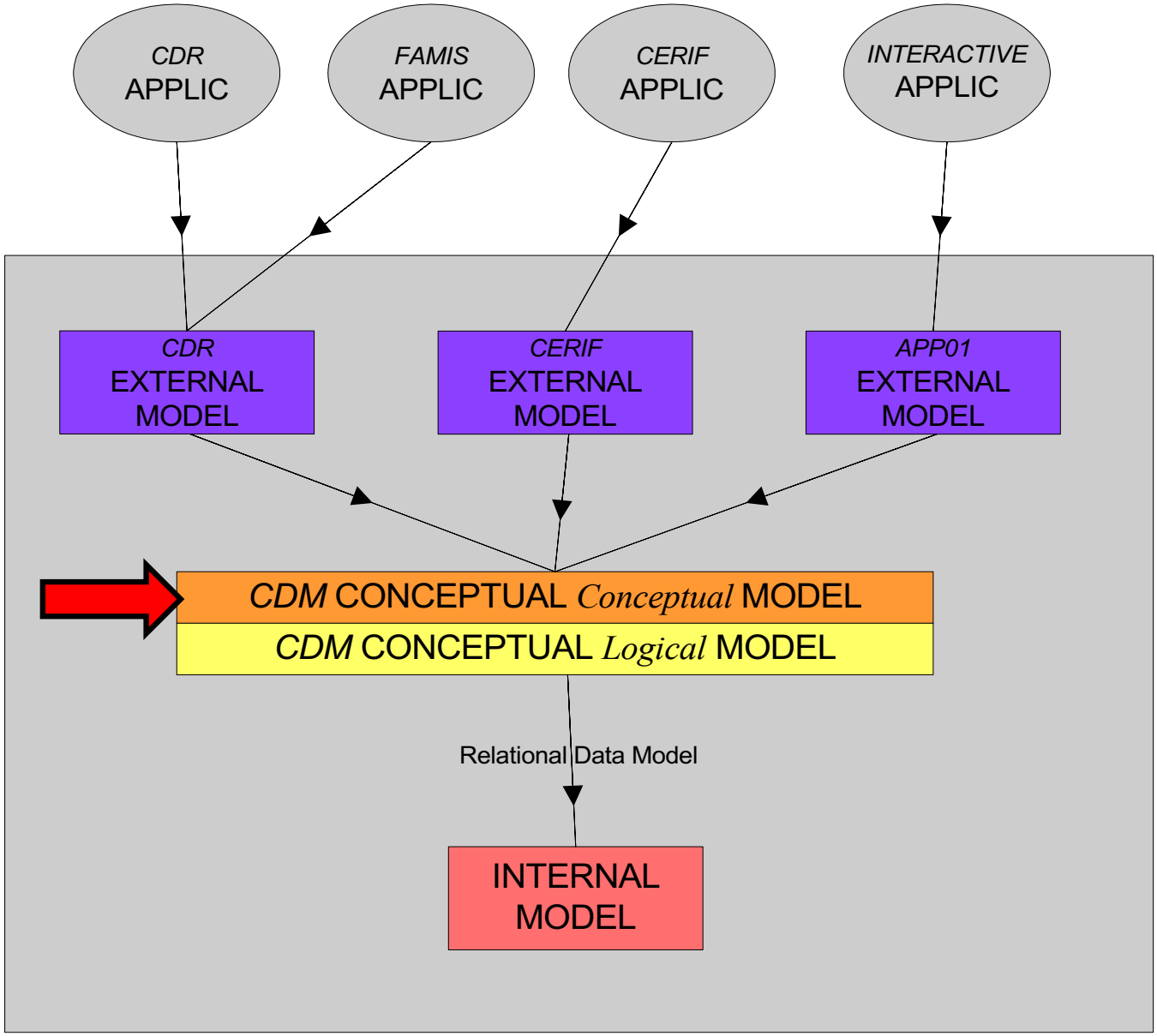
CDM Architecture



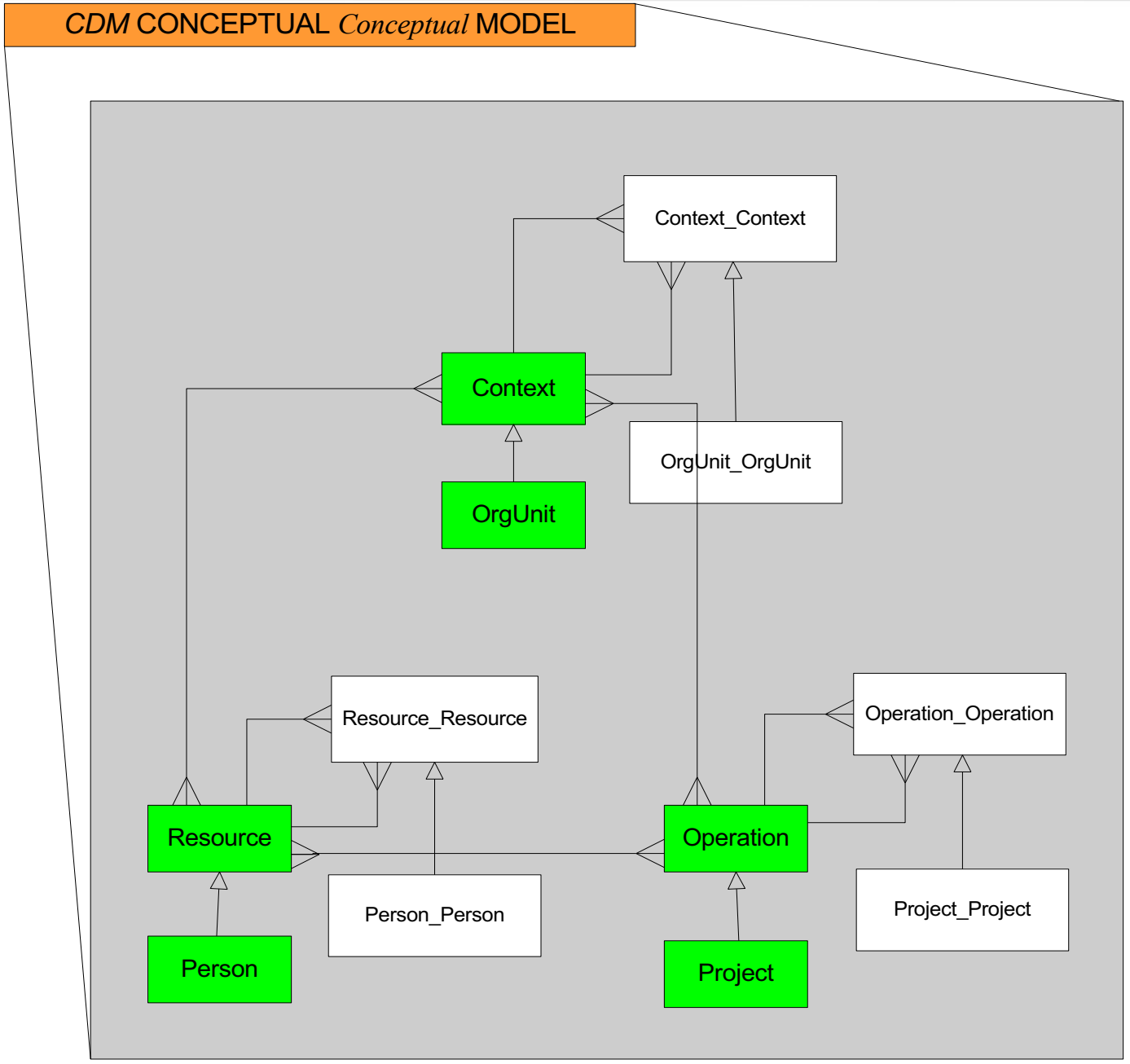
CDM Conceptual Model *Sub-levels*



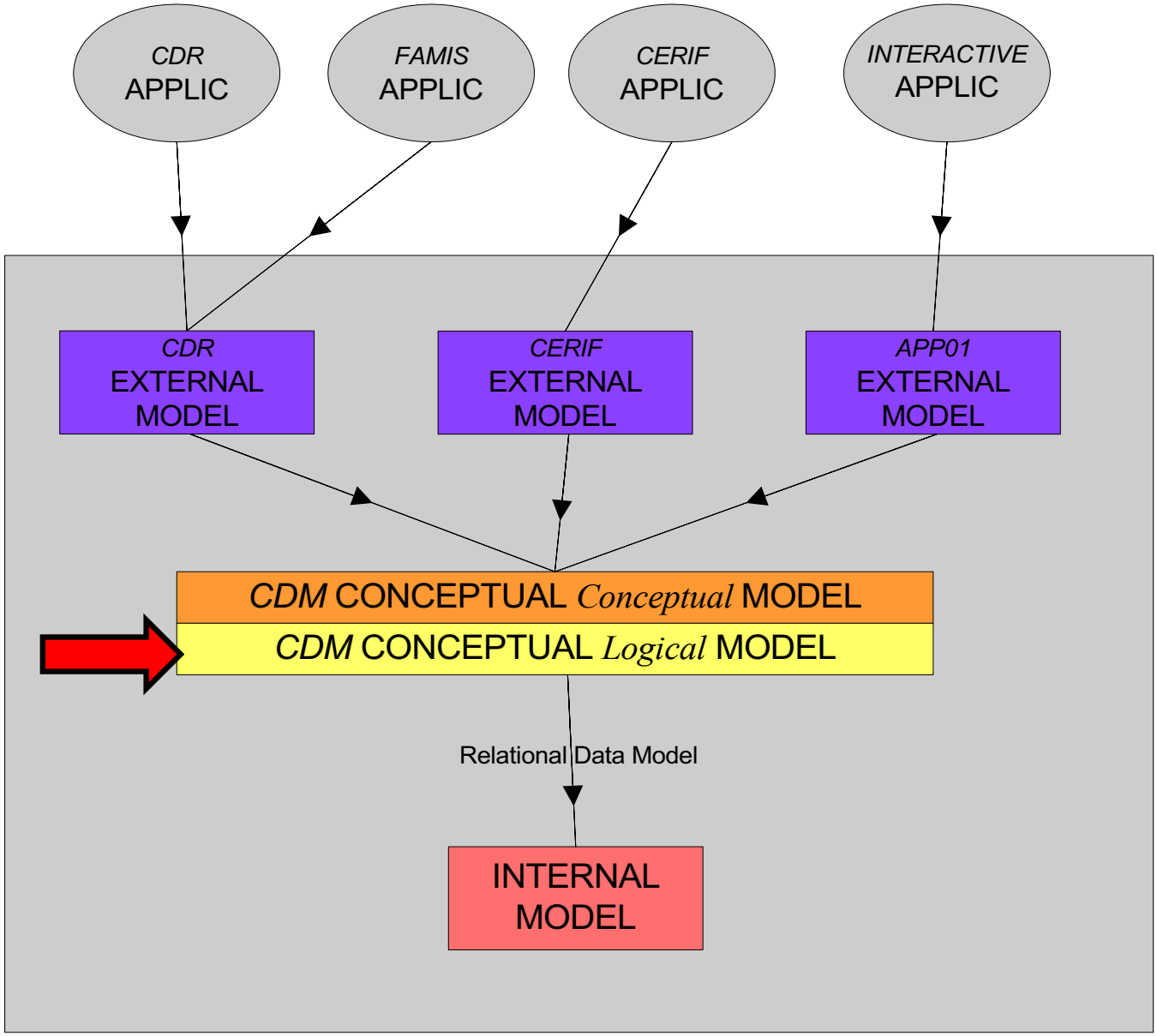
CDM Conceptual *Conceptual* Model



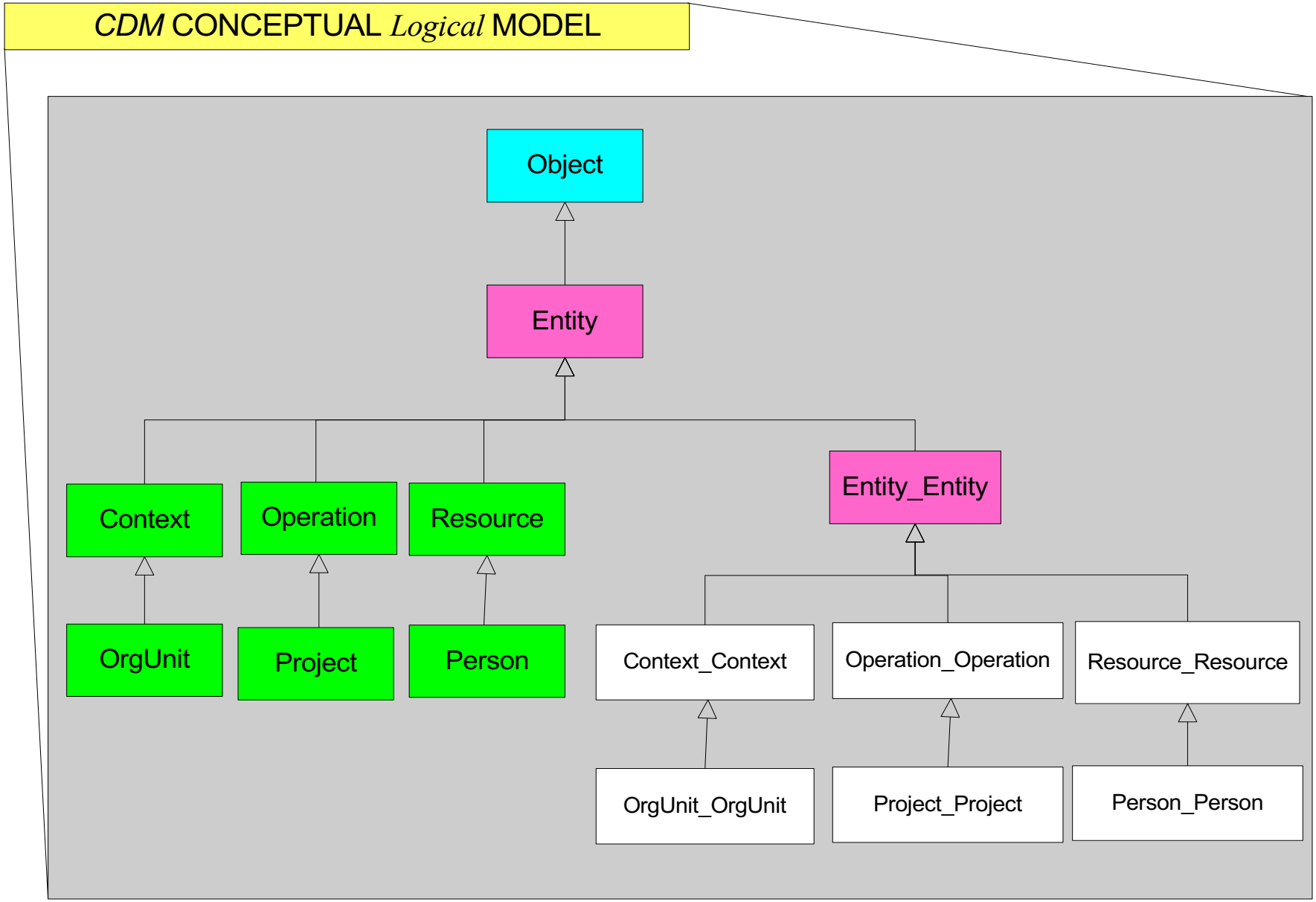
CDM Conceptual *Conceptual* Model



CDM Conceptual *Logical* Model



CDM Conceptual *Logical* Model

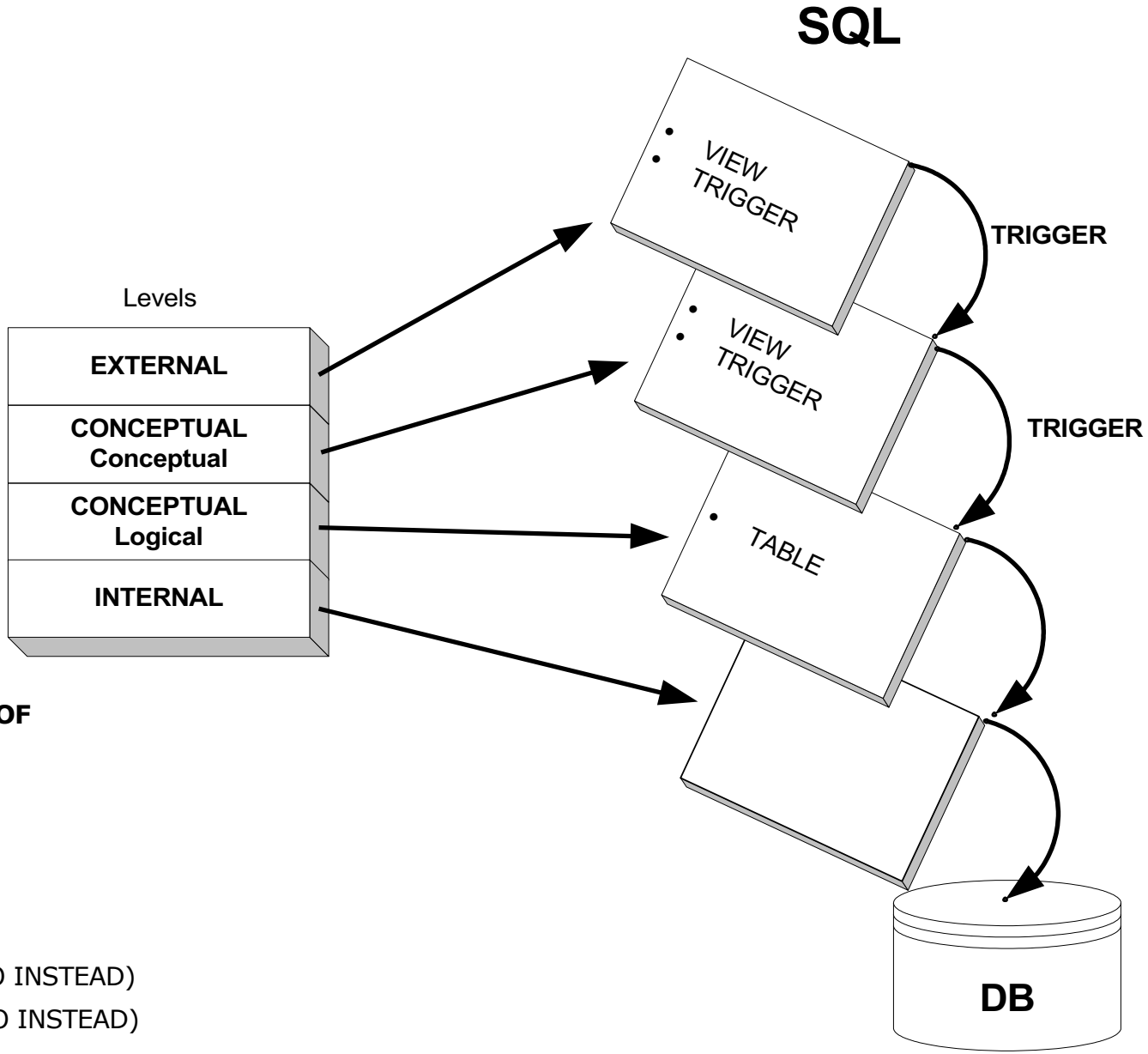


CDM ANSI/SPARC Architecture

The ANSI/SPARC layers provide a degree of data model independence:

- The External layer is relatively independent of the Conceptual *Conceptual* layer.
- The Conceptual *Conceptual* layer is relatively independent of the Conceptual *Logical* layer.

CDM Implementation



CREATE TRIGGER … INSTEAD OF

Supported By:

- Oracle
- Microsoft SQL Server
- IBM DB2
- IBM Informix
- CA Ingres (CREATE RULE … DO INSTEAD)
- PostgreSQL (CREATE RULE … DO INSTEAD)

CCLRC

Temporal CDM

Temporal CDM

Many databases only store *current* facts.
What if we want to store and query *historical* facts?

For example:

- *What was the organisation hierarchy on 1 April 1995?*
- *What was an employee's surname before her most recent marriage?*
- *Which manager had departmental signing powers on 24 November 2004?*
- *What position did this senior researcher hold at CCLRC when he wrote his 1995 journal paper?*

CDM Entity Time

Valid Time - *the period over which a fact was believed to be true in the real world.*

Our *beliefs* may change - so may valid time.

Example: *King Canute lived from 995 to 1035.*

erPerson

ObjId	ObjDomId	EntId	EntDomId	ValTimeBegin	ValTimeEnd	PerFamilyNames	PerFirstNames
1000100	9000000	1000081	9000000	01 Jan 995 00:00	12 Nov 1035 00:00	Canute	King
1000101	9000000	1000083	9000000	01 Apr 1995 09:00	18 Sep 2002 09:00	Smith	Mary
1007000	9000000	1000083	9000000	19 Sep 2002 09:00	NULL	Jones	Mary
1000107	9000000	1000089	9000000	01 Apr 1995 09:00	NULL	Baker	Benjamin John
1000099	6000000	1000089	9000000	01 Apr 1995 15:00	NULL	Baker	Benjamin

CDM Object Time

Transaction Time - *the period over which a fact was recorded to exist in the database.*

Our *records* do not change – neither does transaction time.

Example: The fact "*King Canute lived from 995 to 1035*" was recorded to exist in the database from 7 *Mar 1995* until *now*.

ccPerson

ObjId	ObjDomId	ObjTimeInsert	ObjTimeUpdate	ObjTimeDelete	EntId	EntDomId	PerFamilyNames	PerFirstNames
1000100	9000000	07 Mar 1995 09:00	NULL	NULL	1000081	9000000	Canute	King
1000101	9000000	20 Mar 1995 09:00	NULL	NULL	1000083	9000000	Smith	Mary
1007000	9000000	19 Sep 2002 09:00	NULL	NULL	1000083	9000000	Jones	Mary
1000107	9000000	01 Apr 1995 09:00	04 Apr 1995 11:00	NULL	1000089	9000000	Baker	Benjamin John
1000099	6000000	01 Apr 1995 15:00	02 Apr 1995 10:00	NULL	1000089	9000000	Baker	Benjamin
1000101	6000000	01 Apr 2005 10:00	NULL	02 Apr 2005 12:00	1000089	6000000	Doe	John

Conclusion

- CCLRC
- Corporate Data *Muddle*
- Corporate Data Repository (CDR) Requirements
- Corporate Data Model (CDM) Design
- CDM Architecture
- Temporal CDM